GenRad

GENRAD

2511/2514

VIBRATION CONTROL SYSTEM

DIAGNOSTIC MANUAL

Form No. 2511-0190
Revision 3
June 1986

GenRad, Inc.
Structural Test Products Division
510 Cottonwood Drive
Milpitas, CA 95035

Telphone: (408) 946-6960
Twx: 910-338-0291

LIST OF EFFECTIVE PAGES

Revision 1:	December 1982
Revision 2:	October 1983
Change 1:	June 1985
Revision 3:	June 1986

Page Number	Revision Level
Title	3
A	3
i thru iv	3
1-1 thru 1-2	3
2-1 thru 2-4	3
3-1 thru 3-2	3
4-1 thru 4-2	3
5-1 thru 5-2	3
6-1 thru 6-42	3
7-1 thru 7-8	3
8-1 thru 8-14	3
9-1 thru 9-16	3
10-1 thru 10-4	3
11-1 thru 11-10	3
12-1 thru 12-10	3
13-1 thru 13-8	3
14-1 thru 14-4	3

TABLE OF CONTENTS

Section			Page
1	INTRODUCTION		
	1.0	Introduction	1-1
	1.1	XXDP+	1-1
	1.2	RT-11 Based Diagnostics	1-1
2	XXDP+ USE		
	2.0	XXDP+ Operation	2-1
	2.1	Obtaining a Directory	2-1
	2.2	Loading and Running Programs	2-2
	2.3	Chain Mode Operation	2-2
	2.4	Running a Chain	2-2
	2.5	ENABLE Command	2-3
	2.6	HELP Command	2-3
	2.7	XXDP+ Monitor Errors	2-3
3	RT-11 BASED DIAGNOSTICS USE		
	3.0	RT-11 Based Diagnostics	3-1
4	VERIFICATION		
	4.0	2511/TEST 2514 Chain Programs	4-1
	4.1	Calling Sequence	4-1
	4.2	Movement of System	4-1
5	SAMPLE RUN		
	5.0	Sample Run of 2511.CCC	5-1
6	XXDP+ TESTS		
	6.0	VKAACO LSI-11 Basic Instruction Test (11/12)	6-1
	6.1	VKABA0 LSI-11 EIS Instruction Tests (11/12)	6-2
	6.2	VKACC1 LSI-11 Instruction Tests (11/12)	6-4
	6.3	VKADC0 LSI-11 Traps Test (11/12)	6-6
	6.4	DZKEDA KE11-B Diagnostic	6-8
	6.5	VKAMAD LSI-11 4K System Exerciser	6-9
	6.6	ZKMFA0 MOS/CORE 0-124K Exerciser	6-12
	6.7	LSI-11/73 Diagnostics	6-15
	6.8	ZRXAE0 RX02 Formatter Program	6-16
	6.9	VDLAB0 DLV11-J Test	6-17
	6.10	DLVTST-DLV11-J Test	6-22
	6.11	DLV11-DLV11-J Test	6-22
	6.12	VKAFD1-DRV11 Test	6-23
	6.13	VMSAA0 0-2 MEG Memory Exerciser	6-25
	6.14	JKDBD0 11/23 CPU Diagnostic	6-28
	6.15	JKDAD1 Memory Management Diagnostic	6-30
	6.16	JKDCB0 Floating Point Diagnostic #1 JKDDB0 Floating Point Diagnostic #2	6-34
	6.17	Floppy Disk Performance Exerciser ZRXDB0	6-36
	6.18	Winchester Diagnostics (10MEG and 20MEG Drives)	6-39

TABLE OF CONTENTS

Section			Page
7	LIVE		
	7.0	Usage	7-1
	7.1	Routines	7-1
	7.2	Tests	7-2
	7.3	Starting and Stopping Line	7-2
	7.4	Information Sent to CRT	7-4
	7.5	NPR Test	7-4
	7.6	PROM Board Multiplier Test	7-4
	7.7	Computation Tests	7-5
	7.8	Display Curl Pattern	7-5
	7.9	Display Memory Test	7-5
	7.10	Display Bit Error Identification	7-6
	7.11	Display Chip Error Identification	7-6
	7.12	SUN Test	7-6
8	ADS Tests		
	8.0	Purpose	8-1
	8.1	Software Requirements	8-1
	8.2	Hardware Requirements	8-1
	8.3	Reference Material	8-1
	8.4	Floppy Disk Version	8-1
	8.5	General Information	8-1
	8.6	ADSD1A Tests	8-2
	8.7	ADSD1B Tests	8-2
	8.8	ADSRW - ADS Read Write Program	8-3
	8.9	RTCTS1 - Real Time Clk Test	8-3
	8.10	FDTST - Frame Done Test	8-4
	8.11	SAMPD - Sample Done Interrupt Test	8-4
	8.12	CHTST 'X' - 'X' Channel Sampled Data Test	8-5
	8.13	CDTST - Continue Done Test	8-7
	8.14	TRIGA, TRIGB, TRIGC - Channel X Trigger Test	8-8
	8.15	TRIGE - External Trigger Test	8-9
	8.16	DIST - Harmonic Distortion Test	8-10
	8.17	FSTST - Frame Size Test	8-11
	8.18	SRTST - Sample Rate Test	8-11
	8.19	TCTST1 - Transient Capture Test	8-12
	8.20	DIGTRG - Digital Trigger Test	8-13
	8.21	CHSEL - Selector Test (Option)	8-13
9	RT11 STANDALONE TESTS		
	9.1	Phase Matching Tests	9-1
	9.2	MEMTST	9-2
	9.3	QDAS Tests	9-4

TABLE OF CONTENTS

Section			Page
10	XXDP+ GUIDELINES		
	10.0	XXDP+ Naming Conventions	10-1
	10.1	XXDP+ Media Packaging Guidelines	10-3
11	XXDP+ UPDATE PROGRAM		
	11.0	Introduction	11-1
	11.1	DIR Command	11-1
	11.2	CLR Command	11-2
	11.3	LOAD Command	11-2
	11.4	DUMP Command	11-3
	11.5	XFR Command	11-3
	11.6	SAVE Command	11-3
	11.7	MOD Command	11-3
	11.8	CORE Command	11-4
	11.9	LOCORE Command	11-4
	11.10	HICORE Command	11-4
	11.11	DELETE Command	11-4
	11.12	ZERO Command	11-5
	11.13	BOOT Command	11-5
	11.14	SAVM Command	11-5
	11.15	REN Command	11-5
	11.16	DRIVER Command	11-6
	11.17	PIP Command	11-6
	11.18	READ Command	11-6
	11.19	TYPE Command	11-6
	11.20	DO Command	11-7
	11.21	ASG Command	11-7
	11.22	PATCH Command	11-7
	11.23	EXIT Command	11-7
	11.24	COPY Command	11-8
	11.25	UPD2 Errors	11-8
12	XTECO - TEXT EDITOR		
	12.0	Introduction	12-1
	12.1	XTECO Command Description	12-1
	12.2	TEXT, EDIT, and TECO Commands	12-2
	12.3	XTECO Edit Commands	12-5
	12.4	Buffer Pointer Positioning Commands	12-5
	12.5	Editing Command String Syntax	12-6
	12.6	Command Arguments	12-6
	12.7	Text Type-Out Commands	12-7
	12.8	Deletion Commands	12-7
	12.9	Insertion Command	12-8
	12.10	Output Command	12-8
	12.11	Search Commands	12-8
	12.12	XTECO Errors	12-9

TABLE OF CONTENTS

Section			Page
13	HELP TEXT		
	13.0	XXDP+ Help Text File	13-1
	13.1	XXDP+ Monitor	13-1
	13.2	XXDP+ Utilities	13-2
	13.3	Miscellaneous Operations	13-5
	13.4	XXDP+ Naming Conventions	13-6
14	SUPERVISORY DIAGNOSTICS		
	14.0	Operating Instructions	14-1
	14.1	How to Run	14-1
	14.2	Details of Commands and Syntax	14-3

Section 1

INTRODUCTION

1.0 Introduction

This diagnostic package is a group of tests designed to check the LSI family computer and options, the GenRad hardware, and other vendor supplied hardware. The package is subdivided into two basic divisions:

XXDP+ (2511-0690)
RT11 Based Diagnostics (2511-0691)

1.1 XXDP+

The XXDP+ package is a multi-purpose group of diagnostics supplied by DEC. They have been modified for GenRad systems.

1.2 RT-11 Based Diagnostics

Separate test routines to verify operation of the ADS, QDAS and GenRad 2501 microprocessor are written in TSL and callable from the RT11 operating system as subroutines of TSD25A.

The tests are supplied on a floppy diskette with a double density DY boot. These tests are also used for calibration of the analog and digital portions of the ADS.

This page intentionally left blank.

Section 2
XXDP+ USE

2.0 XXDP+ Operation

Start the execution of the XXDP+ Monitor by using the appropriate bootstrap program. The monitor will print a message identifying itself and requesting that the current date be entered. An example of this message is given below for the DKDP+ monitor.

```
CHMDYC0 XXDP+ DY MONITOR
BOOTED VIA UNIT 0
ENTER DATE:  (DD-MMM-YY):
28K NON-UNIBUS SYSTEM
```

After the date has been accepted by the monitor, the restart address of the monitor is printed.

2.1 Obtaining a Directory

To obtain a directory of the system device, type one of the following:

D(CR) To obtain directory on console terminal, or

D/F(CR) To obtain shortened or "fast" directory on console terminal

C DIRECT To obtain diagnostic directory on console terminal.

The directory contains the following information:

Filnam, Ext File name and extension assigned.

Length Number of blocks used. Decimal number.

Start Starting block number. Octal number.

Date Date when file was put on medium.

2.2 Loading and Running Programs

2.2.1 Type "R" and the program name. Do not type the extension. However, the actual file must have a BIN or BIC extension. This will load and run the program. To just load the program, type "L" and the program name. Once the program is loaded, type "S" to start the program. Starting addresses for DEC diagnostics have a self starting capability at location 200.

2.2 Loading and Running Programs (continued)

2.2.2 If a typing error is made, press the backspace key for each character you wish to remove from the data previously entered. This must be done before the carriage return is pressed.

2.2.3. To abort the function in progress and return to monitor command mode, simultaneously press the "CTRL" and "C" keys (CTRL-C). There may be a time delay before the command mode is reentered.

2.2.4. To freeze output going to the terminal or line printer, type a "CTRL-S" (XOFF). To allow printing to resume, type a "CTRL-Q" (XON).

2.3 Chain Mode Operation

Chain mode operation consists of the sequential execution of programs without operator intervention. Only programs that have been modified to run in chain mode can be chained. Chainable programs are identified in the directory by a BIC extension.

To run chain mode, the XXDP+ monitor uses an ASCII file (known as a chain file) listing the programs to be run and the number of passes each program should run. This file must be on the system device.

A chain file may be generated by use of the XTECO text editor; the file must have a .CCC extension. The chain file may contain any of the commands supported by the XXDP+ monitor. The commands in the ASCII file are executed in the order in which they are encountered. Comments may be included in the file.

2.4 Running a Chain

To execute a chain file the user types;

C FILNAM(CR)
C FILNAM/QV (CR) or

In the first case the pass count specified in the chain file is used by the XXDP+ monitor to determine the number of passes to execute each program. In the second case the pass count is not used and each program is executed only once; the /QV switch provides a single execution mode of operation or quick verify.

When programs are run in chain mode, the hardware or software switch register are set to 0 or preset for proper test execution.

The XXDP+ monitor prints each command taken from the chain file and then executes the command.

When the last command other than another "C" command has been executed, the XXDP+ monitor terminates chain mode and types a prompt (.). It is ready to

accept another command from the console. If the last command is another "C" comand, the chain mode will continue and the chain file specified by this new "C" command will be used.

If the user wishes to terminate chain mode before its normal termination, he may do so by typing a "CNTL-C". However, the monitor will not abort the chain mode until it receives program control from the program currently running.

2.5 "ENABLE" Command

It is possible to change the monitor's memory of what drive it was booted from by typing "E Unit#". For example, if the boot was from drive 0 and the operator types "E 1", then all subsequent monitor commands will cause the medium on Drive 1 to be accessed.

2.6 "HELP" Command

For a complete help file describing DEC monitor and utility operation, Type C HELP1 (CR).

To obtain a HELP message for an explanation of the GenRad package, type: H (CR).

2.7 XXDP + Monitor Errors

? Bad value	- A non-numeric value was entered as a number
? Cksum err	- Checksum error occurred during a load.
? Lookup error	- Non-existent file or device drive.
? Odd Adr	- Invalid address. Must be an even address.
? RD-Dev Err	- Device error on input device.

This page intentionally left blank.

Section 3

RT-11 BASED DIAGNOSTICS USE

3.0 RT11 Based Diagnostics

Diagnostics essential in verifying GenRad hardware are written in TSL callable routines which run under RT11 and TSD25A. These test routines require operator intervention and are explained fully in Sections 9-12. Briefly they are:

Name	For
LIVE	2501 Microprocessor Test
ADSD1A	Multi Channel A/D Subsystem
ADSD1B	Multi Channel A/D Subsystem
ADSAL1	Multi Channel A/D Subsystem Alignment Routines
PHASE	Phase Test for Input Channels
MEMTST	0-128K Memory Test
QDAS1	QDAS Tests
QDAS2	QDAS Tests
QDASA	QDAS Tests
QDASB	QDAS Tests
QDASC	QDAS Tests

This page intentionally left blank.

Section 4

VERIFICATION TEST

4.0 2511/2514 Chain Programs

The verification routine is on a the XXDP+ diskette (2511-0690). Programs have been chained for a standard 2511/2514 configurations. If your system is non-standard, it may be necessary to modify the existing chain or build a new one by using the XTECO text editor.

4.1 Calling Sequence

To begin, simply type C 2511 (CR) for 11/2, C 2514 (CR) for 11/23, or C 2514T (CR) for 11/73 after "." prompt. It will be necessary to occasionally toggle the CRT into "Page Scrol" mode as some tests will flip the CRT into "Page" mode.

4.2 Movement of System

These programs should be run each time the 2511 is moved. Run time is approximately 28 minutes. An example of an error free run is shown in Chapter 5, Section 1.

This page intentionally left blank.

Section 5

VERIFY SAMPLE RUN

5.0 Sample run of 2511. Operator responses are underlined.

.C 2511 (CR)

```
;                              2511 VERIFICATION TESTS
;                              TOTAL RUN TIME IS 28 MINUTES
;
;
;LSI 11/2 BASIC INSTRUCTION TEST

R VKAA??/5

 END PASS
 END PASS
 END PASS
 END PASS
 END PASS
;BASIC INSTRUCTION TEST GOOD.

;EXTENDED INSTRUCTION TEST (EIS).

R VKAB??/5

END PASS
END PASS
END PASS
END PASS
END PASS
;EIS TEST GOOD

R VKAC??/5

DVKACC - LSI-11 FIS  INSTRUCTION TEST
END PASS
END PASS
END PASS
END PASS
END PASS
;FIS TEST GOOD.

;LSI 11/2 TRAPS TEST.

R VKAD??/5

CVKADCO LSI-11 TRAPS TEST

DIBOL INSTRUCTION SET NOT PRESENT
```

```
END OF PASS
;TRAPS TEST GOOD.

;

;2501 EAE EMULATION TEST.

;RUN TIME 8 MINUTES.

R DZKE??/1

EAE OK!
;LSI 11/2 SYSTEM EXERCISER.

R VKAH??/1
DVKAH-A
MEMORY= 017776

;SYSTEM EXERCISER GOOD.

;

;28K WORDS MEMORY TEST.  TO CHECK ABOVE 28K RUN TEST 'MEMTST' ON

;RT-11 DIAGNOSTIC DISKETTE

;

;RUN TIME FOR THIS TEST IS ABOUT 5 MINUTES.

R VMSAA0/2

CVMSAA

MEMORY MAD:

FROM 000000 to 157777

NO PARITY REGISTERS FOUND

PROGRAM RELOCATED TO 00000000

END PASS #    1
PROGRAM RELOCATED TO 00000000

END PASS #    2  .5K RESTORED
;28K MEMORY TEST GOOD.
;
;END OF 2511 TEST SEQUENCY FOR XXDP+ DIAGNOSTICS.  TO TEST THE
;DISK USE DISKETTE P/N 2511-0692.  TO TEST THE ADS, 2501 MICO CPU
;AND Q-DAS USE THE DISKETTE P/N 2511-0691.  SEE DIAGNOSTIC MANUAL
;FOR INFORMATION ON THESE AND OTHER RELATED TESTS.
;
;
```

Section 6

XXDP+ TESTS
(2511-0690)

6.0 VKAAC0 LSI-11 Basic Instruction Test (11/2)

6.0.1 Program Description

This program tests all the basic instruction of the LSI-11 (except trap-type) which includes control chip, data chip, microms, and other circuitry on the LSI-11 CPU module. Trap diagnostic should also be run to make sure that the basic LSI-11 is functional. This diagnostic does not make a pass with T-bit set.

6.0.2 Equipment

LSI-11 Standard Computer and 4K of memory.

6.0.3 Storage

Program Storage - The routines use memory 0 - 17500.

6.0.4 Starting Address = 200

6.0.5 Error Reporting

On finding an error the processor will come to a halt after placing the error number in location 100. In most cases the comments beside the halt tell what was being checked. In some cases the test can get to a halt via two ways:

1. Wrong Test Sequence
2. Error in Actual Test

6.0.6 Error Recovery

Restart at 200.

6.0.7 Execution Time

Execution time of the diagnostic is less than second. First "end pass" will be typed out within a second and every consecutive "end pass" will be typed out within 20 seconds.

6.1 VKABA0 LSI-11 EIS Instruction Tests (11/2)

6.1.1 Program Description

This program tests the LSI-11 extended instruction set (ASH, ASHC, MUL, and DIV) option using registers 0-5 at least once with each instruction. It is also checked that extended instructions can be interrupted (by the console teletype). The program should be run for at least 2 passes with all switches low. It will be required to have a SLU with TTY registers having addresses of 176560-66 and interrupt vectors of 60 for receiver and 64 for transmitter.

6.1.2 Equipment

LSI-11 standard computer with EIS option and 4K of memory.

6.1.3 Storage

Program Storage - The routines use memory 0 - 17500.

6.1.4 Starting Address

After loading the program it should always be started at 200.

LTC switch must be in 'off' position.

6.1.5 Switch Settings

A 16 bit location at address 422 has been used to give the following options by inserting a 1 in their respective positions.

Bit	Octal Value	Function
15	100000	Halt on Error
13	020000	Inhibit Printout

All typeouts can be suppressed by making bit 13 of address 420 high, in other words by placing a 20000 in location 420.

6.1.6 Error Printout

The format is as follows:

ADR ERRNM

WHERE:

ADR = Address of Error
ERRNM = Error Number

In most cases the comment beside the call for halt subroutine tells what was being checked and what was expected. All printouts will be suppressed when bit 13 of location 420 is high.

6.1.7 Error Recovery
Restart at 200.

6.1.8 Execution Time

Due to the random characteristic of the interrupt tests, the execution time can be 5 seconds or more per pass. However, normally first "end Pass" will be typed within 5 seconds and within 50 seconds for every consecutive 10 passes.

6.2 VKACC1 LSI-11 FIS Instruction Test (11/2)

6.2.1 Program Description

This program tests the LSI-11 floating instruction set (FADD, FSUP, FMUL, and FDIV) option with fixed number patterns, using each register at least once as the stack pointer. It also checks stack overflow and that the floating instructions can be interrupted (by the console teletype).

6.2.2 Equipment

LTC switch must be in off position to run diagnostic.

6.2.3 Storage

Program Storage - The routines use memory 0 - 17500.

6.2.4 Starting Address

The program should always be started at 200.

The program will loop and "END PASS" will be typed after completion of every pass.

A minimum of two passes should always be run.

6.2.5 Operational Switch Settings

All bits 0 except SW 11 is worst case testing. With bit 11 of location 422 high each subtest will be looped upon until completion of 256 passes of that subtest. "END PASS" will be typed upon completion of a pass of the entire program. Alternate pass will run with the T-bit set.

Switch Settings Are - A 16 bit location at 422 has been used to give the following options by inserting a 1 in their respective positions.

BIT #	OCTAL VALUE	FUNCTION
15	100000	Halt on Error
14	040000	Scope Loop
13	020000	Inhibit Printout
12	010000	Inhibit Trace Trapping
11	004000	Enable Iterations of Subtest
10	002000	Bell on Error
09	001000	Loop on Error
08	0004XX	Loop on Test in Bits 7 Through 0

6.2.6 Error Printout

The format is as follows:

ERRNM ADR PS SP ANS1 ANS2 ANS3 ANS4 ANS5 ANS6

Where:

ERRNM	=	Error Number
ADR	=	Address of Error HLT
PS	=	Processor Status
SP	=	Contents of Stack Pointer Register
ANS1-6	=	Error Data Read from the Stack(s). From 0 to 6 of these may be typed depending on the number following the HLT; e.g., HLT+3 would type ANS1 through ANS3, HLT (by itself) would stop after ERRNM, ADR, PS, and SP.

6.2.7 Error Recovery

Restart at 200.

6.2.8 Execution Time

Due to the random characteristics of the interrupt tests, the execution time can be half a minute or more. However, normally "END PASS" will be typed within 40 seconds with all switches down. Execution time will increase by a factor of 256 when iterations of each subtest are enabled.

6.3 VKADC0 LSI-11 Traps Test (11/2)

6.3.1 Program Description

This is a test of all operations and instructions that cause traps, oddities of register 6, interrupts, the reset and wait instructions.

The program checks that on all trap operations register 6 is decremented the correct amount, that the correct PC is saved on the stack, that the old condition codes and priority are placed on the stack and that the new status and condition codes are correct. Both the "trap" and "EMT" trap instructions are tested to see that all combinations will trap. Checked also are the RTT and the RTI instructions, and that all restricted instructions will trap. Verification of the "BPT" instruction (00003) which is used for software debug routines ODT, DDT, is done. Also, the trace bit is checked to see if it causes a trap. Special checks are made to see if bus error traps occur on non-existent memory.

6.3.2 Equipment

PDP-11 or LSI-11 standard computer with a SLU unit and 4K of memory. The clock must be disabled.

6.3.3 Setup Procedure

The program starts at 200. If it is desired to reset the pass count back to zero start at locations 210. The clock must be disabled for this program to run correctly.

6.3.4 Switch Register Bits

The operator has the following options by setting the software switch register (location 422).

Bit 6=1 (100 octal) to suppress testing EIS and FIS opcodes (70000-75037) for reserved instruction trapping in the last test of this diagnostic.

Bit 5=1 (40 octal) if we want to suppress "End of Pass" typeout.

Bit 4=1 (20 octal) to suppress testing opcodes 75400-76777 for reserved instructions traps in the last test of this diagnostic.

Bit 3=1 (10 octal) to suppress testing opcodes 170000-177777 for reserved instructions traps in the last test of this diagnostic.

Bit 2=1 (4 octal) to suppress testing opcodes 76030-76057 (DIS reserved opcode space) nor EIS opcodes for reserved instruction traps in the last test of this diagnostic.

Bit 1=1 (2 octal) if there is no I/O device in the address space 160000-167777. If this bit is set, this address area will be checked by test 70, expecting it to all trap, or all not.

6.3.5 Errors

All errors will cause a halt. The PC+2 of the halt instruction is printed on the console device by the LSI-11's U-ODT.

The program checks to see that the P.C. doesn't jump within the tests, by a sequence count called "$TSTN". This test is a sequential increment and compare count.

```
EX:     CODE

INC     @#$TESTN          ;Update Test Number
CMP     #N,@#STESTN       ;Sequence Error?
BNE     SOME LOCATION     ;Branch to Error Halt on Seq Error
```

Important

If an error is detected, it could be because of two reasons.

Wrong Test Number
Error in the Present Test

6.3.6 Execution Time

For one pass approximately 8 seconds; then it types "END OF PASS" approximately every 2 minutes.

6.4 DZKEDA KE11-B Diagnostic (Use for GenRad 2501 Microprocessor)

6.4.1 Abstract

This is a diagnostic package for use with the KE11-B or GR 2501 microprocessor. There are three parts to the main test, these being Register Tests, Logical and Arithmetic Shift Tests, and Multiply and Divide Tests.

6.4.2 Requirements

1. PDP 11 (4K minimum)
2. KE11-B Interface or 2501 microprocessor.
3. Console Terminal

6.4.3 Control Software Switch Settings

NOTE: Only available when starting at address 210.

SR0-1	Skip logical and arithmetic shift tests
SR1=1	Skip multiply and divide tests

6.4.4 Start address is 200.

6.4.5 Pass Definition

Normal operation is with all switches down. Under these conditions a pass will require approximately 8 minutes at which time "EAE OK!" is printed on the console terminal.

6.5 VKAHA1 LSI-11 4K System Exerciser

6.5.1 Abstract

This is an LSI-11 4K systems exerciser. It is a test of the processors ability to operate peripherals in interrupt mode. It is not a complete test of the peripherals themselves.

The test occupies less than 4K of memory but does run a memory address test of all available memory to 28K.

A processor instruction test is run while allowing the peripherals to interrupt at random. If relocation is enabled the processor instruction test is run in each 4K memory bank.

6.5.2 Equipment

LSI-11 with minimum 4K of memory and console output device.

6.5.3 Program storage = 0000-17476

6.5.4 Start/Restart Address = 200

6.5.5 Switch Settings

Software Switch Register = Location 176
Location 176 = 114167 when loaded.

"Static" switch settings can only be modified if the test is to be restarted at 200.

"Dynamic" switch settings can be modified during a CTRL-C halt and the test continued.

Dynamic Switch Settings

BIT15	(100000)	-Halt on Error
BIT14	(040000)	-Loop on Subtest
BIT13	(020000)	-Inhibit Error Printouts
BIT12	(010000)	-Inhibit T Bit Trapping
BIT11	(004000)	-Inhibit Sub Test Iteration
BIT10	(002000)	-Inhibit Processor Instruction Test
BIT 9	(001000)	-Inhibit Instruction Test Relocation
BIT 8	(000400)	-Restart Program on Error
BIT 7	(000200)	-Inhibit End of Pass Printout

Static Switch Settings

BIT 6	(000100)	-Inhibit ASR33 Low Speed Reader Test (Do not use)
BIT 5	(000040)	-Inhibit Line Printer Test (Do not use)
BIT 4	(000020)	-Inhibit DRV11 Parallel Line Unit Test (Do not use)

Static Switch Settings (continued)

BIT 3 (000010) - Inhibit EIS/FIS Test
BIT 2 (000004) - Inhibit Console Output Test (Do not use)
BIT 1 (000002) - Inhibit Floppy Unit 1 Test (Do not use)
BIT 0 (000001) - Inhibit Floppy Unit 0 Test (Do not use)

6.5.6 Operator Action Required

6.5.6.1 Load the program into memory using standard procedure for PDP-11 binary formatted programs.

6.5.6.2 If I/O Devices Selected: (Not applicable on 2510 MicroModal)

6.5.6.3 Set software switch (location 176) to desired options. If an option is selected and its address fails to answer, the test will halt at PC = 0602.

NOTE Loc 176 =114167 when the program is first loaded.

6.5.6.4 Start the test at 200. If the test halts at PC = 602 recheck switch options in location 176 and restart.

6.5.6.5 The following printouts will occur:

"DVKAH-A"
"Memory= XXXXXX"
Where "XXXX" is the highest memory location found by sizing.

NOTE: If relocation inhibited or running in memory will not be sized and "Memory = 17776" will be printed.

6.5.6.6 End of pass will be indicated by the following printout.

"PASS= 00001 ERROR= 00000 RXERROR= 00000 TIME= 00000"

6.5.7 CNTRL-C Interrupt Option

The test can be interrupted at any time by typing CNTRL-C. The test will halt at PC=172. The switch options (location 176) can be modified and the program continued by typing "P".

6.5.8 Error Printouts

Error Printout Format:

"PC = 000000 PSW = 000000 RETURN = 000000"

Where:
PC = The program counter where error found.
PSW = Processor status at time of error
RETURN = The beginning address of the instruction subtest being executed. The "return" address can be used as the beginning address for a loop on subtest.

6.5.9 Execution Time Table

Typical Test Times: (In form - Minutes:Seconds)

7:30 = 4K + All options + All I/O devices.
5:25 = 4K + No TBIT + Iterations + All I/O devices.
1:25 = 4K + No TBIT + No Iterations + All I/O devices.
0:50 = 4K + No Options + No I/O devices.

6.5.10 Pass Printout Format

"PASS = Total No. of Passes Loc. "PASS" (524)

Where:
PASS = Total No. of Passes Loc. "PASS" (524)
ERROR = Total No. of errors. Loc. "ERROR" (526). This is a running total of all errors. It is useful to get an error count when error printouts are inhibited.
RXERROR = Floppy total soft error count. Unit 1 and Unit 0. Loc. "RXSOFT" (516). The floppy test is retried 10 times before it is considered a hard error.

While the soft error count gives no detailed error description, if it is no 0000 the floppy diagnostic should be run.

6.6 ZKMAF0 MOS/CORE 0-124K EXERCISER

6.6.1 Abstract

This diagnostic will test 0 - 124K of MOS or core memory on any PDP-11 family computer. Some tests are worst case for MOS and some for CORE, but all tests are always run. The tests occupy less than 2K of memory so it can be used to test a system with only 4K of memory. If only 4K exists, however, the absolute loader is not saved.

6.6.2 Getting Started

6.6.2.1 Start at 200.
The memory limits will be printed.

6.6.2.2 Restart.
To halt the test, type CNTRL-C, this will insure the program is relocated back to lower memory. Be patient, the CNTRL-C is only recognized at the end of the current subtest.

6.6.2.3 Software switch setting is location 176.

6.6.3 Switch Setting

BIT15(100000)	Halt on Error
BIT14(040000)	Loop in Subtest Defined by Bits 0-3
BIT13(020000)	Inhibit Error Printouts
BIT12(010000)	Enable Testing Above 28K (with memory management)
BIT11(004000)	Enable Parity Testing
BIT10(002000)	Halt After Each Subtest
BIT09(001000)	Inhibit Program Relocation
BIT08(000400)	Type First Failing Bit Error Per 4K
BIT07(000200)	Enable Long Galloping Test
BIT06(000100)	Inhibit Memory Sizing
BIT05(000040)	Inhibit "PASS#XX" Printouts
BIT04(000020)	Inhibit Printouts
BIT03-BIT00	Beginning Test Number

6.6.4 Equipment

Standard PDP-11 family computer with a console output device and from 4K to 124K of memory. Program will also run on 30K LSI systems.

6.6.5 Storage

Program storage: 0000 - 7744. Program expands for error history and to save absolute loader or XXDP chain monitor.

6.6.6 Long Gallop Option

Normal worst case SR setting = 0000. For long gallop SR = 200. Long gallop option should only be used if an MOS memory problem is suspected and no other subtests will fail. The test time is increased 25 times.

6.6.7 Error Message Format

The error printout consists of 6 octal words in the following format:

"LOCATION GOOD BAD PC ERROR PASFLG"

"ADR ERR" will be printed prior if an addressing error is suspected.
"PAR ERR" will be printed prior if a parity error trap occurred
!CAUTION! if parity error the good data printout is the parity module unibus address that failed.

Where:

LOCATION	=	Failing Memory Location
GOOD	=	Good Data (Data that was expected)
BAD	=	Bad Data (Data that was found)
PC	=	Program Counter at Error Call
ERROR	=	Failing Error No.
PASFLG	=	Contents of Location PASFLG. This may not be relevant.

The test will continue after the error printout. "NO KT" will be typed if testing above 28K selected and no memory management is found.

"NO PAR" will be typed if parity option selected and no parity modules were found.

6.6.8 Error History

Locations in memory are set aside to collect a history of the failing bits in a particular memory bank. This data is collected for every error regardless of switch settings.

Normally the data is output at the end of testing, but if CNTRL-C is typed, it is output at the end of the current test.

The error history is intended to highlight whether the errors are due to one bit failing or only address errors.

Error History Format:

Error Bank Count

Where:

ERROR=	Bit that failed (number of the failing bit in decimal I.E., 0-15 will be typed out of the words "ADR ERR" or "PAR ERR" will be typed out if address error or parity error was seen in the specific bank of memory.
BANK =	4K memory bank in which this failure was seen. A 0 for 0 to 4K, a 1 for 4 to 8K and so on.
COUNT=	Number of times this memory bank failed. (377 is maximum failure count recorded.)

6.6.9 Restrictions

Memory under test should be contiguous. For systems having non-contiguous memory the memory boundaries should be defined by the operator. (Contiguous memory is defined as a memory that can be both read and written in consecutive locations.)

6.6.10 Execution Time

Here are some typical execution times.

LSI-11 and 4K = 100 seconds.
LSI-11 and 8K = 5 minutes.
LSI-11 and 28K = 14 minutes.

6.7 LSI-11/73 Diagnostics

6.7.1 Tests Available

ZKDJA1	11/73 CPU Test
ZKDKA1	11/73 Memory Management Test
ZKDLA0	11/73 Floating Point Test
ZKDMA0	11/73 Cache Memory Test

6.7.2 Starting Procedures

All tests have a starting address = 200. Type R filename for continuous run. Each test requires less than 5 seconds a pass.

6.7.3 Special Considerations

The Cache Memory Test is hard programmed for an 18-bit address bus. i.e., SWR = 400 Setting bit 9 will extend the test time considerably.

6.7.4 Chain Test 2514T.CCC

The 11/73 tests are Chained together in the file 2514T.CCC. Run this program by typing C 2514T.

6.7.5 Errors

Each test will type error information to the CRT.

6.8 ZRXEA2 RX02 Formatter Program (2511-0690 Diskette)

6.8.1 Abstract

This program is intended to format a floppy disk to double density.

6.8.2 Hardware

1. Any PDP-11 Processor
2. RX02 Compatible Subsystem

6.8.3 Storage

This program requires at least 4K words of core. If loading via XXDP+ more storage will be needed for the XXDP+ monitor.

6.8.4 Starting Procedure

Start at location 200(8) to initialize registers.

6.8.5 Operation

The program operation in "user mode" is simple. After initial start the program will ask the operator if help is wanted and will type a short description of the program if answered yes. The program will also ask if a full diskette scan is wanted to verify CRC. If address modification is selected then the program will ask the operator to enter the bus address and vector address.

6.8.6 Conditions of Operation

The program expects to run on an RX02 or DSD440interface and RX02 or DSD440 drive subsystem that have passed all diagnostic tests. Errors are reported, but printouts are not extensive enough to diagnose hardware failures.

6.9 VDLAB0 DLV11-J Test

6.9.1. Program Purpose

This diagnostic is a logic test to verify the operation of the DLV11-J serial interface.

Testing is done in two phases:

1) All selected channels per DLV11-J module are tested individually

2) The DLV11-J module is tested as a whole for channel interaction problems. This diagnostic is designed to test and detect errors to the logic level (not to the chip level).

The operator must install data wrap around connectors to do data testing. To bypass data tests, the operator must modify the user switch register at address 1220, see program options in paragraph 6.9.5.

The default addresses and vectors are as follows:

177560 - Console Interface Device Address
176500 - First serial channel address of up to 8 consecutive serial line devices.

60 - Vector for console device interface.
300 - Vector for first of 16 devices.

This program is designed to run on any Q-BUS PDP-11 with 4K of memory and a DLV11-J (Q-Bus) module.

6.9.2 System Requirements

Hardware Requirements:

Any Q-Bus PDP-11 family processor
4K memory - minimum
A special data wrap around connector or equivalent (required if data wrap around tests desired)

If channel 3 is configured as the console:
Tests 6-12, 14-17, 21, 22 are bypassed.

If data wrap around tests are bypassed:
Tests 7-12, 14-17, 21, 22 are bypassed.

Software Requirements:

This diagnostic is designed to run in any of the following ways:

Stand alone
With XXDP monitor+ (chainable if renamed to .BIC extension)

6.9.3 Assumptions

This diagnostic assumes that the operator has initialized location 176 (Switch register), 1220 (options), 1250 (Address) and 1244 (Vector) to the proper values.

6.9.4 Operating Instructions

Loading and Starting Procedures.

Use standard procedure for PDP-11 absolute binary formatted media.

There are 2 starting locations for this diagnostics:

1) Location 200 for all normal testing starts and restarts.

2) Location 400 for interrupt vector debug... offline only.

6.9.5 Switch Register

1) Type control G (^G); This will allow the TTY to enter data into location 176 at selected points within the program.

2) The test will then type: SWR=XXXXXX is the octal contents of the software switch register.)

3) After the 'NEW=' has been typed then the operator can do one of the following at the CRT:

   a) Type a number to be loaded into location 176 followed by a (CR). (Only numbers between 0-7 will be accepted.) Leading zeros need not be typed, and if more than 6 digits are typed the last 6 will be used. If a (CR) is the first key depressed the software switch register contents will not be changed.

   b) If a control U (^U) is depressed then the program will send you back to Step 3.

   c) If the input character is not one of the characters mentioned above then a question mark (?) will be typed followed by a carriage return and a line feed sequence then proceed from Step 3 (erasing all previous input).

      The diagnostic will continue on typing (CR).

      Note: Because of frequent bus resets in the program, it may be necessary to do 'CONTROL-G' several times. Alternately, a 'BREAK' and manually loading location 176, followed by a 'P' to proceed will also work.

Software Switch Register Options (SWREG)

BIT	15	SET = 100000	=	Halt on Error
	14	SET = 40000	=	Loop on Test (To be used only while testing in progress)
	13	SET = 20000	=	Inhibit Error Typeouts
	12	SET = 10000	=	Enable Performance Reports
	11	SET = 4000	=	Inhibit Iterations
	10	SET = 2000	=	Bell on Error
	9	SET = 1000	=	Loop on Error
	8	SET = 400	=	Loop on Test in SWR(7:0)
	7:0	=	=	Number of Test to Loop On (Used with bit 8) (All tests previous to the selected test are executed first with 1 iteration only)

6.9.6 Program Options

This program requires the address of the first RCSR (stored at 1250) and its intettupt vector (stored at 1244). It wil be able to address any DLV11-J starting at the specified base address thru 2 consecutive modules.

Examples:	1250	176500	(Default)	GR 175640
	1244	300	(Default)	GR 340

Location 1220 is used as a bit map to indicate which unit numbers are present and will be tested.

Bit Position	Definition	Default Value
0	# of Data Bits Transmitted 0 = 7 Bits, 1 = 8 Bits	1 = 8 Bits
1	Parity Enabled	0 = No
2	Even Odd Parity	0 = Odd
3	Break Detection Enabled	1 = Yes
4	Run Data Wrap Around Tests	0 = No
(11:9)	Console Device	1 = Yes = Console on Module 1, Channel 3

(11:9) = 0 = Console Not on DLV11-J
(11:9) = 1 = Console On Module 1
(11:9) = 2 = Console On Module 2

6.9.7 Execution Times

Execution times for an LSI-11 processor with one DLV11-J module at shipment configuration:

	Ch. 0,1,2	At 9600 baud.
	Ch. 3 (Console)	At 300 baud.
Are:	First Pass-	30 Sec.
	Additional Passes	90 Sec.

The test time is baud rate dependent; higher baud rates result in shorter pass times.

6.9.8 Error Information

Since this diagnostic was designed to fit in 4K of memory the error typeout is very brief. The format of the error typeout is as follows:

Test #_____, Error #_____, PC=_____, Address=_____, Vector=_____

Where all values typed are octal. The address and vector refer to the failing channel. For further information the listing must be consulted. Bits 15, 13, 10 and 9 of the switch register control the sequency of events after an error is caught.

Bit 15 Set: Causes the program to halt in the error routine. If the program is continued, it will proceed from where it halted.

Bit 13 Set: Disables the printing of the error message.

Bit 10 Set: Causes the bell to ring on error.

Bit 9 Set: Causes the diagnostic to loop from beginning of test to error.

The only halt in this diagnostic is in the error routine, and is executed only if Bit 15 of the switch register is set when an error occurs.

6.9.9 Performance and Progress Reports

Performance Reports. (Bit 12 set in the switch register)

As each channel (4 channels/DLV11-J) completes one pass of the diagnostic the following will be typed:

'CSR: ——' : The base address of the line under test
'Vector: ——' : The associated vector
'Errors:——' : The total number of erros on this device on this pass

After all modules and channels to be tested have been exercised, an end pass statement is typed:

"END PASS#——."

6.9.10 Summary of tests and special subrountines

Test 1 Addressability

This test verifies that all 4 registers of the channel under test respond to their addresses.

The following 3 tests test all 'READ WRITE' bits

Test 2 BREAK - TCSR 0 SET, CLEAR, RESET

Test 3 XMITIE - TCSR 6 SET CLEAR, RESET

Test 4 REVRIE - RCSR 6 SET, CLEAR, RESET

Test 5 XMITRDY - TCSR 7 - IS SET BY INIT

Test 6 XMIT RDY - TCSR 7 - Clears when TBUF is loaded with a character and that it sets witnin a reasonable amount of time.

Test 7 Outputting a character from TBUF (with wrap around connected) results in RCVRDONE setting within a reasonable almount of time and that reset clears the bit.

Test 10 RCVRDONE is cleared by setting reader enable

Test 11 RCVRDONE is cleared by reading RBUF

Test 12 Overrun and Error Bit - RBUF 14

Test 13 Transmitter interrupt logic test

Test 14 Receiver interrupt logic test this test covers all of the receiver side of the interrupt logic in character mode.

Test 15 Test data wrap around; FLAG MODE

Test 16 Test data wrap around: INTERRUPT MODE

Test 20 Not a test - send back to loop

Test 21 Test that channels interrupt at assigned priority

Test 22 Test data transfers with all active lines interrupting

6.10 DLVTST - DLV11-J Test (No test plug required)

6.10.1 Purpose

This diagnostic will test the DLV11-J serial inteface as configured for the GenRad 2511.

6.10.2 Operating Instructions

See Section 6.9 (VDLAB0 DLV11-J).

6.10.3 Program Modifications

The following changes were made to VDLAB0.

Address	From	To	Comment
1220	1031	11	No console or wrap plug
1244	300	340	Vector charge
1250	176500	175640	Address change
4730	500	5	Timing loop bug

6.10.4 Special Requirements

None. The test will run without the maintenance test plug connected.

6.11 DLV11 - DLV11-J Test (with test plug)

6.11.1 Purpose

This diagnostic will test the DLV11-J serial interface as configured for the GenRad 2511.

6.11.2 Operating Instruction

See section 6.9 (VDLAB0 DLV11-J)

6.11.3 Program Modifications

The following changes were made to VDLAB0.

Address	From	To	Comment
1220	1031	11	No console or wrap plug
1244	300	340	Vector change
1250	176500	175640	Address change
4730	500	5	Timing loop bug

6.11.4 Special Requirements

Test plugs must be installed on all four ports. The test will loop data via the transmitter and receiver logic.

6.12 VKAFD1 - DRV11 Test

6.12.1 Program Purpose

This is a logic test of the DRV11, to allow testing of the data lines and interrupts. A special maintenance cable (RC08R) is used by default.

This test will operate on one DRV11. Special operational procedures are required to operate on other than the primary DRV11.

6.12.2 Requirements

Equipment

LSI-11

DRV11

Test Cable (BC08R)

6.12.3 Starting procedure

200 - Normal entry to test one device

If any program options are required, set the appropriate bit in the software switch register at location 422.

Start program at 200.

Program will print "END OF PASS" following each pass.

6.12.4 Control switch setting

This program contains a software switch register for optional selection. For it to operate, the operator must select the appropriate option by setting or resetting the respective bit in the word.

To do this, the LSI-11 must be in ODT mode.

Starting address or addresses

200 = Start of Test -- For normal testing

The program will cycle continuously unless halted by the operator or some error condition.

To halt the program, depress the halt switch. ODT will display the PC at which it was halted.

Reset at this time.

Continue the program via a "P" or a "G" command.

Software switch settings (Address 422)

Bit15	-	Continue on error	(100000)
Bit14	-	Loop on current error	(040000)
Bit13	-	Not used	(020000)
Bit12	-	Not used	(010000)
Bit11	-	Not used	(010000)
Bit10	-	Loop on current test	(002000)
Bit9	-	Run test module (inhouse only)	(001000)
Bit8	-	Inhibit wrap cable	(000400)
Bit7	-	Not used	(000200)
Bit6	-	Not used	(000100)
Bit5	-	Not used	(000040)
Bit4	-	Not used	(000020)
Bit3	-	Not used	(000010)
Bit2	-	Not used	(000004)
Bit1	-	Not used	(000002)
Bit0	-	Not used	(000001)

Selection of test options

This test will run with the wrap cable by default. To inhibit testing with the wrap cable, the operator must set Bit8 in the switch register (location 422).

Wrap cable

The wrap cable is required to test transfer of data into and out of the input buffer, and the device interrupts.

This diagnostic is approximately 95% effective when run with the wrap cable, and approximately 60-70% effective when run without it.

6.12.5 Execution time

Typical run times (one pass)

Quick verify	1 Sec.
With wrap cable	10 Sec.

6.12.6 Error reporting

All error reports will be done via a halt within the program. This will cause ODT to display the PC+2 of the error halt.

In order to continue, the operator must issue a "P" to continue the program, or may set the error loop switch prior to continuing.

6.13 VMSAA0 0-2 MEG Memory Exersizer

6.13.1 Program purpose

This program has the ability to test memory from address 000000 to address 17757777. It does so using:

A) unique addressing techniques,
B) worse case noise patterns, and
C) instruction execution throughout memory.

The intent of this program is to test as comprehensively as possible all mos memories used on the LSI-11 Bus without concentrating on any one system. Although the tests relate to general designs they may be complete for certain systems. This test is also not intended to be a total 100% test of the memory. Other tests that do I/O may find memory problems that this test is unable to.

6.13.2 System Requirements

LSI-11/2, LSI-11/23 Bus family processors.
Minimum of 16KW of memory.

Optional....
Any parity memory control module.
KTF11 memory management.

6.13.3 Diagnostic Prerequisites

Before running this program, a CPU diagnostic should be run to verify the functionality of the processor and PDP-11 instruction set.

For LSE-11/23:CJKDB?? DIAG
For LSE-11/2 :CVKAA?? DIAG

If memory management is to be used, then the KTF11 diagnostic CJKDA?? should also be run before this program.

6.13.4 Operating Instructions

Starting address 200:

Normal program execution.

Starting address 204:

Restart program using previously selected parameters.

6.13.5 Program Options

The software switch register (location 176) is used for all operational switch settings.

The user can type CTRL G (^ G) to allow SWR changes during program execution.

SW15 = 1 or up....	Halt on Error
SW14 = 1 or up....	Loop on Test
SW13 = 1 or up	Inhibit Error Typeout
SW12 = 1 or up....	Inhibit Memory Management (initial start only)
SW11 = 1 or up....	Inhibit subtest iteration (not used)
SW10 = 1 or up....	Ring Bell on Error
SW9 = 1 or up....	Loop on Error
SW8 = 1 or up....	Loop on Test in SWR(4:0)
SW7 = 1 or up....	Inhibit Program Relocation
SW6 = 1 or up....	Inhibit Parity Error Detection
SW5 = 1 or up....	Inhibit Exercising Vector Area (Locations 0-1000).

6.13.6 Execution Times

Execution time is dependent on type of memory, and amount of memory. Worse case run times with MOS memory are:

A. For non-parity memory

Full pass:	< 5 minutes for 64KW 30 minutes for 1280 KW

6.13.7 Error Reporting

There are a total of 31(8) types of error reports generated by the program. Some of the key column heading mnemonics are described below for clarity:

PC = Program counter of error detection Code. (V/PC=P/PC)

V/PC = Virtual program counter. This is where the error detection code can be found in the program listing.

P/PC = Physical program counter. This is where the error detection code is actually located in memory.

THP/PC = Physical program counter of the code which caused a trap.

MA = Memory address.

REG = Parity register address.

PS = Processor status word.

IUT = Instruction under test.

S/B = What contents should be.

WAS = What contents was.

6.13.8 Error Halts

With the 'HALT ON ERROR' switch (SW15) not set there are several programmed "HALTS' in the program:

A) In the error trap service routine for unexpected traps to vector 4. This one will occur is a 2nd trap to 4 occurs before the error report for the first has had a chance to be printed out.

B) In the reloction routine if the program is being relocated back to the first 8K of memory and the program code was not able to be transferred properly.

C) In the case of error reporting and there is no terminal to allow the information transfer.

D) In the power fail routine if the power up sequence was started before the power down sequence had a chance to to complete itself.

E) In the memory mapping routine or any of the address control routines, failures to find a meaningful map.

6.14 JKDBD0 11/23 CPU Diagnostic

6.14.1 Abstract

This program is a combined version of the three basic 11/34 diagnostic programs with modifications and enhancements made to account for the differences between the two processors.

The program contains three parts: CPU, TRAP and EIS tests. In the first and second parts, the program will halt on error. In part three, EIS test, when an error is detected, the error PC and error number will be typed, then the program will continue execution.

PART ONE: CPU test, this is the first part of the main program. This test checks out the basic PDP-11 instructions in all addressing modes with various types of data patterns.

PART TWO: TRAP test, this is the second part of the main program. This is a test of all operations and instructions that cause traps. Also tested are trap overflow conditions, oddities of register 6, interrupts, the reset and wait instructions. This program checks that on all trap operations register 6 is decremented the correct amount, that the correct PCS is saved on the stack, that the old condition codes and priority are placed on the stack and both the "TRAP and "EMT" trap instructions are tested to see that all combinations will trap. Checked also is that all reserved instructions will trap. The trace bit is checked to see if it causes a trap. The RT1 and RTT instructions are checked. Stack overflow is also checked for all the trap instructions.

Special checks are made to see if BUS error traps occur on non-existent memory. All instructions that are reserved should trap to location 10, and the PC that points to the trapping instruction should be placed on the stack.

PART THREE: This program tests the extended instruction set < ASH, ASHC, MUL, and DIV> using registers 0-5 at least once with each instruction. This program tests all the EIS instructions of the 11/23 for ASH and ASHC instructions every even pass is executed with destination mode 0 for all registers and every odd pass with destination mode of 67. The diagnostic does not make a pass with T bit set.

6.14.2 Starting Procedures

The program is started by loading address 200. The restart address is 1024. Program identification will be typed after the first pass of the whole program.

6.14.3 Error Handling

In parts one and two, all errors will cause a halt.

The program checks to see that the PC doesn't jump erratically within the tests.

If an error is detected, the program will halt. It could be because of two reasons.

a) Wrong test number (sequence error)
b) Error in the present test.

In part three, any error, includes sequency check error will cause the error message to be types. The program will continue execution after type out.

The error reporting format is as follows:

ERROR! PC AND ERROR # ARE:

PC #

ERROR #

6.14.4 Switch Settings

Since no handware switch register is available, the program automatically uses the contents of Loc. 176 as the software switch register. The initial content of Loc. 176 is 000000, the user may pre-set this location before starting the program.

BIT #	OCTAL VALUE	FUNCTION
15	100000.....	HALT ON ERROR
13	020000.....	INHIBIT ERROR PRINTOUT
2	000004.....	PROGRAM RESERVED -- PROGRAM WILL SET IF CIS OPTION IS AVAILABLE
1	000002.....	30K SYSTEM DO NOT CHECK FOR TRAPS BETWEEN 28K-30K
0	000001.....	SKIP TRAPS TEST

6.14.5 Execution Times

The run time for a single run (the first pass) is one second. After the first pass, the program will iterate every 15 times before the end of pass message is typed again. The run time for each additional end of pass message typed is approximately 15 seconds.

6.15 JKDAD1 MEMORY Management Diagnostic

6.15.1 Abstract

This program was designed using a "BOTTOM UP" approach starting with the smallest segment of memory management logic possible and building to cover all of the logic.

The program begins by testing some of the internal CPU data and address paths and address detection logic, then works outward through the memory management registers. After the registers are found to be useable, relocation (construction of physical addresses from a virtual address and the associated PAR/PDR information) is tested followed by testing of the abort and status segments of logic. Finally, checks of special abort sequences and testing of the MFPI/MTPI instructions are done.

A 11/23 processor with a minimum of 16K of memory and a console terminal are required to run the program.

6.15.2 Preliminary Programs

Before this memory management diagnostic is run, the following CPU diagnostic should be run:

JKDBD0 11/23 CPU TESTS

Also, one of the main memory diagnostics should be run to scan at least the first 16K to see that a program can be executed.

6.15.3 Starting Procedures

The program is started by loading address 200. Since there is no hardwae switch register, the program will use the software switch register at location 176 (location 174 will be used as the software display register). In that case, the program will ask for the initial switch register value by typing "SWR = XXXXX NEW = " after typing the name of the program (XXXXXX = THE OCTAL CONTENTS OF LOCATION 176).

6.15.4 Control Switch Settings

Switch	Octal/Value	Use
SW15	100000	HALT ON ERROR This switch when set will halt the processor when an error is detected after the error message has been typed. Pressing continue will resume testing (see section 3.1 about loading the switch register before continuing).

6.15.4 Control Switch Settings (continued)

Switch	Octal/Value	Use
SW14	040000	LOOP ON TEST This switch when set will cause the program to loop on the current subtest.
SW13	020000	INHIBIT ERROR TYPEOUTS This switch when set will inhibit the typing of error messages.
SW12	010000	INHIBIT TRACE TRAP This switch when set will inhibit T-bit trapping which normally takes place during every other pass starting with the third pass.
SW11	004000	INHIBIT SUBTEST ITERATIONS This switch when set inhibits iterations of each subtest after the first pass. If this switch is not set, each subtest is run 200. times.
SW10	002000	BELL ON ERROR This switch when set will ring the console terminal bell when an error has been detected.
SW9	001000	LOOP ON ERROR This switch when set will cause the program to loop on the first failure which is encountered even if the failure is intermittent.
SW8	000400	LOOP ON TEST IN SWR <7:0> This switch when set will cause the program to loop on the test whose test number is set in bits 7-0 of the switch register.

6.15.5 Loading The Switch Register

To load the software switch register while the program is running, a control G (^G) should be typed on the console terminal. The original value of the software switch register will be requested.

6.15.5 Loading The Switch Register (continued)

In response to A ^G or at the beginning of the program, the program will type:

SWR = XXXXXX NEW =

Where "XXXXXX" is the current octal contents of Loc. 176. the operator may then type any one of the following:

XXXXXX<CR>	One to six octal digits followed by a carriage return which will be loaded as the new value for the switch register
<CR>	Just a <CR>, leaves the switch register as it is.
XXX ^U	A control-U (^U) will cause al of the digits typed so far to be ignored.
^C	Will cause the program to type the present test and pass numbers, request a new value for the switch register, and jump to the end-on-pass routine so the program will go directly to the next pass with a new switch register value.
<ILL. CHAR>	Any character typed which is not any of the above or an octal digit will cause the program to type a "? <CRLF>" and react as though a ^C has been typed.

Note: Recognition of a ^G may be hampered by execution of a couple "reset" instructions within the program.

6.15.6 Execution Times

The run time for a single pass with no iterations or trace trapping is approximately 5 seconds.

6.15.7 Error Information

If an error is detected, the program will trap to the error handling routine. The value of bits 15, 13, 10, and 9 in the switch register are considered in reporting an error. The error information will be typed unless SW13 - 1.

If SW15 - 1, the processor will halt after the error is reported. If the contents of the software switch register are to be changed, a ^G should be typed before pressing "CONTINUE" to resume testing.

6.15.7 Error Information (continued)

Below is an example of an error which could have occurred during execution of the program:

```
MEM. MGMT. REG. BITS NOT ST CORRECTLY
REGISER WROTE READ READ-(BINARY)
ADDRESS (OCTAL) (OCTAL) 543210987654321   TESTNO  ERRORPC
177572     040000    060000   0110000000000000  000012    022060
```

6.15.8 End-of-Pass Message

At the end of each pass of the program the pass number and total number of errors since the last end-of-pass are reported in the end-of-pass message. For example:

```
END OF PASS #2  TOTAL ERRORS SINCE LAST REPORT 0
```

6.16 JKDCB0 Floating Point Diagram #1
JKDDB0 Floating Point Diagram #2

6.16.1 Abstract

The two programs JKDCB, JKDDB are designed to detect and report logic faults in the F-11 MMU and floating point chip set. The design is an attempt to reach all micro-code locations.

Note that error reports in these programs are based upon the knowledge that all previous tests (CPU, MMU) have been run and in most cases that there is only a single point fault exists. If the programs or tests are not run in order then error messages may not be accurate.

6.16.2 Starting Procedure

Load address 200

Set console switches (if console is present)

Press start

On first pass the program will identify itself. Note that if there is no physical console the program will request the operator for initial value for the software switch register.

The program will loop and an end of pass will be typed at the end of every pass.

6.16.3 Operational Switch Settings

The switch settings are:

	Octal	
SW<15>=1...	100000	HALT ON ERROR
SW<14>=1...	40000	LOOP ON CURRENT TEST
SW<13>=1...	20000	INHIBIT ERROR TYPE OUTS
SW<12>=1...	10000	INHIBIT T-BIT TRAPPING
SW<11>=1...	4000	INHIBIT ITERATIONS
SW<10>=1...	2000	RING TTY BELL ON ERROR
SW< 9>=1....	1000	LOOP ON ERROR
SW< 8>=1....	400	LOOP ON TEST SPECIFIED IN SW<6>

6.16.4 Errors

SUMMARIES

When an error is encountered, an error message accompanied by the error PC are typed. There are four standard error messages used, describing the probable cause of failure, such as: probably bad MMU chip; bad DP1 chip; bad hybrid FP chip; floating point error.

6.16.4 Errors (continued)

ERROR RECOVERY

SW<15:9>=0...	Most errors will cause execution to go to the start of the next test after the message is typed. A few tests are in sections. In these tests an error will cause execution to go to the next section after the message is typed.
SW<15>=1...	The program will halt after typing the error message. Pressing the console continue will cause the program to continue as if SW<15>=0.

6.16.5 Execution Times

Less than 2 seconds for each program on any pass.

6.17 Floppy Disk Performance Exerciser ZRXDB0

6.17.1 This program exercises one RX02 compatible subsystem (two drives), maintains drive statistics and provides run summaries so that seek and data error rates may be determined. The performance exerciser will give the user confidence, after running successfully, that the system is performing within specification.

6.17.2 Hardware Questions

The following series of questions comprise the parameters necessary to identify each floppy disk subsystem.

RX BUX ADR -
This parameter defines the base bus address for the floppy disk subsystem. Type a carriage return.

VECTOR ADR -
This parameter defines the interrupt vector address for the floppy disk subsystem interface. Type a carriage return.

DRIVE # -
This parameter defines the floppy disk subsystem drive number (0 - 1). Type a 0.

EXP WRD-TYPE -
Type a carriage return.

6.17.3 Software Questions

EXERCISE # -
Enter # from table
0 = Default to 7
1 = Write only
2 = Write/read
3 = Write/read/data check
4 = Read/data check only
5 = Read only (CRC check)
6 = Write/read/data check on alternating drives
7 = Write/read/data check +/read/data check

DATA PATTERN # -
Enter # from table
0 = Default to 7
1 = Zeros
2 = Ones
3 = Floating zero
4 = Floating one
5 = 125
6 = 333
7 = Random

TRACK SEQUENCE # -

Enter # from table

0 = Default to 7
1 = Increment O.D. up to I.D.
2 = Decrement I.D. down to O.D.
3 = Increment O.D., then decrement I.D.
4 = Bounce between O.D. and I.D.
5 = Bounce between decreasing I.D. and increasing O.D.
6 = Bounce between O.D. and decreasing I.D.
7 = Random

O.D. = Outside diameter (track)
I.D. = Inside diameter (track)

DEVICE FATAL THRESHOLD LEVEL -

The Device Fatal Threshold Level (DFTL) is initially set=1. This threshold level equals the number of hard errors that will cause a device fatal error when the DRS "EVL" flag is set. The "EVL" flag also causes 10 soft errors to be reclassified as a hard error, which if DFTL = 1 becomes a device fatal error.

RUN TEST IN DOUBLE DENSITY -

If test is in wrong density, operator will be asked if the diskette is to be reformatted.

RUN TEST IN DELETED DATA MODE -

If answered yes, deleted data mode will be done first.

ANY PROGRAM CONTROL FLAGS

If answered yes, the following questions will be asked.

Retry on error, log soft + hard errors?

If retry is not set, soft errors are logged as hard errors.

Recalibrate or seek errors?

Print only 10 data errors + continue?

Clear statistical tables before next pass?

MODIFY TRACK ADDRESS LIMITS -

If answering yes, the following is asked:

Outer diameter ADR #?
Inner diameter ADR #?

MODIFY SECTOR ADDRESS LIMITS -

If answering yes, the following is asked:

Min. sector ADR #?
Max. sector ADR #?

RXXX EXPANSION <CR>

Answer with a carriage return.

6.17.4 Error Information

SYSTEM FATAL ERRORS

System fatal errors are used to indicate that an error was detected by the diagnostic supervisor in relation to loading and controlling the diagnostic process.

DEVICE FATAL ERRORS

Device fatal errors are a result of:

1. Reaching a device fatal threshold level ("DFTL"). An ("DFTL") =1 will cause 1 hard error to be classified a device fatal error. This level is initially set=1, but may be modified by the operator.

2. An error that is considered fatal to the device, but testing will continue.

HARD ERRORS

Hard errors are a result of:

1. Ten retries of a soft error or

2. A non-recoverable error

SOFT ERROR

Soft errors are media related errors and if retry on error is set, it is tried up to ten times, then classified as hard errors.

If retry on error is not set, the error is logged as both a soft and hard error.

6.18 Winchester Diagnostics (10MEG and 20MEG Drives)

6.18.1 Tests Available

DEC Standard	GR Modified	Name
ZRLGB0	ZRLCT1	CONTROLLER TEST 1
ZRLHB0	RLCTL2	CONTROLLER TEST 2
ZRLIC0	ZRLDT1	DRIVE TEST 1
ZRLKB1	ZRLKB1	PERFORMANCE EXERCISER

6.18.2 Chain Tests Available

10MTS1	Full Height 10 Meg Winchester
20MTS1	Half Height 20 Meg Winchester

ZRLGB0 CONTROLLER TEST 1

Memory Location	From	To
20644	1404	404
23766	1404	404

ZRLHB0 CONTROLLER TEST 2

Memory Location	From	To
17360	4737	240
17362	14062	240
17364	5737	240
17366	2652	240
17370	1412	240

ZRLICO DRIVE TEST 1

Memory Location	From	To
6662	51524	240
36462	1406	406

ZRLKB1 PERFORMANCE EXERCISER

Memory Location	From	To
24406	10406	100402
24414	104455	12701
24416	1752	4234
24420	2654	5301
24422	5716	1376

6.18.4 Loading and Starting Procedures

All tests are supervisory style tests. Take care in entering the correct drive type and controller protocal.

The full print-out from the above dialogue might look like this (0=OPERATOR, D=DIAGNOSTIC):

```
.R CZRLHB                                   0
DRS LOADED                                  D
DIAG. RUN-TIME SERVICES REV. D APR-79       D
CZRLH-B-0                                   D
CZRLH TESTS WRITE DATA, READ DATA,          D
AND WRITE CHECK OPERATIONS
UNIT IS RL01, RL02                          D
DR>STA/PASS:1/FLAGS:HOE(CR)                 D,O

CHANGE HW (L) ? Y(CR)                       D,O

# UNITS (D) ? 2(CR)                         D,O

UNIT 0                                      D
RL11 (L) Y ? N (CR)                         D,O (N=RLV11)
BUS ADDRESS (0) 174400 ? (CR)               D,O
VECTOR (0) 160 ? (CR)                       D,O
DRIVE TYPE = RL01 (L) Y ? N (CR)            D,O (N=RL02)
BR LEVEL (O) 5 ? (CR)                       D,O
DRIVE (O) 0 ? (CR)                          D,O

UNIT 1                                      D
RL11 (L) Y ? N (CR)                         D,O (N=RLV11)
BUS ADDRESS (O) 174400 ? (CR)               D,O
VECTOR (O) 160 ? (CR)                       D,O
DRIVE TYPE = RL01 (L) ? N (CR)              D,O (N=RL02)
BR LEVEL (O) 5 ? (CR)                       D,O
DRIVE (O) 0 ? (CR)                          D,O
```

6.18.5 Error Reports

Test run errors are reported to the CRT. To obtain a summary report, type 'PRT' at the >prompt on completion of the test.

6.18.6 Performance Summary

Performance reports are given when a drive is dropped or, at operator request (print), the format is:

```
TIME: HH:MM:SS RLCS: XXXXX DRIVE: Y DRIVE TYPE = RLOX
*** RUNNING OR DROPPED DH:DM
PACK SERIAL _#:    315
TOTAL SEEKS:       IIIII
WORDS READ:        JJJJJJJJJ
WORDS WRITTEN:     KKKKKKKKK
```

6.18.6 Performance Summary (continued)

```
ERRORS
DRV-ER:   N     SEEK:   N     TRACK:  N     DATA:   N
HARD:     N     SOFT:   N
DCK:      N     HCRC:   N     NXM:    N     HNF:    N
DLT:      N     OPI:    N
```

Where:

HH	is hours since start/restart
MM	is minutes since start/restart
SS	is seconds since start/restart
XXXXXX	is address of controller
Y	is drive number
DH	is hour at which drive was dropped
DM	is minute at which drive was dropped
I I I I	is total number of seeks since start time 0:00:00
JJJJ	is total number of words read since start time 0:00:00
KKKK	is total number of words written since start time 0:00:00
N	is number of that type error since start time 0:00:00

Maximum errors allowed

Hard	20 per logical unit
Soft	20 per logical unit
Seek	0 in one hour continuous testing per unit.

This page intentionally left blank.

Section 7

LIVE

(2511-0191)

7.0 Usage

LIVE is a diagnostic program written in TSL used for the 2501 systems.

The uses of LIVE include:

1. Part of the RCT for all 2501 systems - validates the general "health" of the system.
2. Use as a board verification test and as an aid in board debugging once the board has been greenlighted by the GR tester.
3. Use as a board quality verification once a board is working. Done by stressing the board with extremes of temperature and voltage while running LIVE.
4. Use as a system debugging aid when the system is basically functional, but appears to run incorrectly or intermittently. This program can sometimes locate the problem to the board at fault, and in some cases to the section or even the chip with the fault.

7.1 Routines

LIVE consists of the following 17 routines:

A. Assembly Language Routines, Callable from TSL

1. TIME Stores and reads the 2501 Real Time Clock
2. CORSZ (Core Size) Determines the # of 4K word blocks in memory
3. MWORD (Move Word) Moves a word to or from any TSL block
4. DISPLY Displays a TSL block
5. V8 Converts decimal to octal.

B. TSL Routines

1. HELP Prints out the test available when using LIVE
2. LIVE This is the main routine from which all tests can be made
3. LIVF (Live Fast) Same as LIVE 8, 0
4. LIVL (Live Long) Same as LIVE 31, 40 (for overnight testing)
5. LIVB (Live Beep) Same as LIVE 31, 40, 0. System beeps twice every third pass.
6. LIV Same as LIVE 12,40 (for heat stressing)
7. PTEST Determines if the next test should be executed
8. PERR (Print Error) Writes error messages to the terminal
7. FMUL (Fractional Multiply) Uses the prom board multiplier
10. RANXY (Random X,Y) Generates a Random TV coordinate
11. LOGIT Logs the status of the test and accumulated errors
12. PBID Prints out the BIB information of a TSL block
13. BIBS Uses PBIB to print out the BIBS of blocks B0 through B15
14. BTIME Prints program statistics after 'STOP KEY' pressed.
15. CIRCLE Displays pattern for CRT alignment.

16. GRID Displays pattern useful for CRT alignment.
17. SUN Displays pattern useful for checking display vector hardware.
18. SVLIVE (Save Live) Stores the above 23 routines on disk.

NOTE: Routines 7 through 14 above are subroutines called by LIVE

All the above routines can be loaded by the following procedure.

```
.R TSL25 or .R TSD25A(CR)
LOAD 'LIVE'
```

7.2 Tests

Now if you type HELP, the following message will be printed out

Test #	Description
1	NPR Test
2	Prom Board Multiplier Test
4	Computational Test
8	Display Curl Pattern
16	Display Memory Test
32	Display Bit Error Identification Test
64	Display Chip Error Identification Test

7.3 Starting and Stopping Live

LIVE is called with three parameters. The first parameter specifies the test you want to run, the second specifies how many of the errors (if any) should be logged on the thermal printer. The third parameter specifies whether or not the system keyboard will emit a double "beep" every third pass of the program. A zero specifies a beep, while a one means no beep. This feature allows the user to run LIVE, and be sure it is still running, without always watching the CRT screen. Thus, if you wanted to run the computational test with no beep only, you would type:

```
LIVE 4, 75, 1
```

That command would start off the system in an infinite loop, continually repeating Test #4. All errors encountered would be logged on the terminal, but only the first 75 would be sent to the printer. If you want to do more than one of the tests, simply add up the desired test number. Thus:

```
LIVE 31, 0, 1
```

would do all the tests (in sequence, in an infinite loop) except the last two tests (related to display error identification). Once called, LIVE asks for a specific time limit for execution:

```
How long? POSITIVE # = MINUTES, NEGATIVE # = HOURS
```

After a number is keyed in, LIVE begins to run. On the very first pass of LIVE x,x,x, a header is displayed. This header lists the version number of LIVE.TSL, the system memory size, the user selected time limit, and the TSL data block BIB information. Then, the tests are executed until one of the following happens:

1. The "STOP" key is pressed
2. The time limit programmed is reached
3. A hardware fault causes the program to stop
4. The power is removed from the system.

The "STOP" key causes the program to cease execution as soon as possible and print a message on the CRT screen. The content of this message will vary - depending on whether or not any errors were encountered during the test. In the following sample printouts, the TSL error #45 is normal.

1) LIVE 31, 40 ,1 - STOPPED AFTER TEST #1 . . .NO ERRORS!

```
PASS NUMBER 3

TIME = 8301 MINUTES (LIMIT = 1 HOUR)

ERR 45
* * * *
610
PTEST
2430
LIVE
110
LIVL
```

2) LIVE 8, 0, 1 - STOPPED AFTER TEST #8 . . .

```
# OF ERRORS = 2
PASS NUMBER 7
TIME = .3189 MINUTES (LIMIT = 30. MIN)

ERR 45
* * * *
610
PTEST
2970
LIVE
110
LIVF
```

Example 1 shows a program that was stopped during the third pass, with a user time limit of one hour. There were no errors detected during the run and the total test time was less than a minute.

Example 2 shows a program that ws stopped during the seventh pass, but did not run successfully. Two errors were detected, (but not sent to the printer) during the test. The total test time was less than one minute, and the user time limit was 30 minutes. This run was aborted after LIVE test #8.

To run a different type of LIVE test at any time, one only has to press "STOP", wait for the message to print, then key in LIVE x,x,x.

If the user time limit is reached during a run, the system stops the test, "beeps" for aobut three seconds, then prints a message on the CRT screen and on the thermal printer. A typical message is:

LIVE 31, 40, 1 - TEST COMPLETE . . .NO ERRORS!

PASS COUNT = 4

ELAPSED TIME = 2.022 MIN

If there were errors detected during a run, the message format would be similiar to the previous one, i.e., it would not say "NO ERRORS!", it would list the number of errors encountered.

7.4 Information Sent to CRT

After each pass (i.e., after executing each test selected once) the following information is sent to the terminal.

1. The number of passes completed
2. The number of accumulated errors since the test was started
3. The elapsed time in minutes
4. The elapsed time in hours

If the number of passes completed is a multiple of 256. All this information is also sent to the printer so a log can be kept for overnight testing.

A more complete description of each test follows.

7.5 NPR Test

Before any of the tests are started, B0, B1, B2, and B3 are defined and their starting addresses are saved in B8 and B9. (B8 contains least significant 16 bits of the address and B9 contains the two most significant address bits). B0, B1 are type 0 TSL blocks and B2, B3 are Type 1 (i.e., in extended memory). B2 is defined at the very start of extended memory and B3 is defined to be the highest 512 words in memory. (CORSZ is used to determine where this is)

The NPR test uses the information in B8, B9 to send a walking ones pattern to the first sixteen words of B0, B1, B2, B3. The NPR capability of the microprocessor is used to do the transfers. The NPR is then used again to read the first sixteen words of the 4 blocks and compares the words read with what they should be. Any errors are logged by PERR on the terminal.

7.6 PROM Board Multiplier Test

First the multiplier is used to multiply 3072 * 4096. The correct answer is 384 and an error will be logged if any other answer is found. Then 64 pairs of random integers are multiplied using the PROM board multiplier, with the result of each multiply being added into a checksum. If the checksum is different than the known correct value an error is logged.

7.7 Computation Tests

The first two tests are written to avoid using the EAE; so, if there is a problem with the EAE emulation it should show up here. First the TSL random number generator is used to fill up B0 with 512 random numbers. Then B0 is fixed and DFT'd and ASPEC'ed into B2. Then the block is IFT'd then DFT'd and ASPEC'ed again into B2. Then B2 is moved to B1 (since B2 is in extended memory) where we can compute a checksum of the result. If the checksum is different than the known correct one, an error is logged. Since this string of operations is very heavy in number crunching, almost all computational type errors or EAE emulation problems will show up here.

NOTE: In future versions of LIVE, the second ASPEC in this test will be changed to include a third parameter so as to check out this new feature. It is not done in this version because an error in the JUL4 version of the microcode prevents that feature from working properly.

7.8 Display Curl Pattern

Block B1 is zeroed, loaded with two constants, DFT'd, and then displayed in the phase plane plot. This quickly produces a curl pattern on the screen. The test is useful because it tells you something about the computational ability of the microprocessor. The NPR ability of the Unibus interface card, and the healthiness of the display all in a very fast quickly moving test.

7.9 Display Memory Test

First 50 random vectors are drawn on the screen. Then the following sequence is done twice:

A. Sequence if this is the only test being executed

The top 54 lines of display memory are read back and stored into block B4. The display is scrolled and a display scrolling checksum is added into 16. Then the 54 lines in B4 are written back at the bottom of the screen (thus accomplishing a circular scroll).

B. Sequence if this is not the only test being executed

Only the top line of the display memory is read into B4, and only that line is written to the display after it is scrolled. This sequence still checks the memory readback feature of the display but not as thoroughly as sequence A. Sequence B is used because sequence A takes too long.

If the accumulated scrolling checksum is different than the known correct value an error is logged.

NOTE: Due to an error in microcode of the JUL4 version (which is what LIVE version 01 is meant to run on) the display scrolling checksum is virtually meaningless, and therefore, this test will not detect any errors except through visual observation of the screen until that problem is corrected in the microcode.

7.10 Display Bit Error Identification

This test is most useful to run when one of the other tests or other visual observations shows errors or instabilities in the display. If the errors are related to a particular bit failure of any of the components of the display board which are hooked onto the 16 bit parallel memory data bus of the display memory, then this test will identify which bit is the culprit. In this test 32 vertical lines are first drawn using bit 0. Then this is redrawn with dotted vectors and redrawn again with dotted vectors in the complement mode. This is repeated with all 16 bits with the bit number being identified on the screen. If one or more of these patterns looks different than the usual ones, the problem is at least somewhat isolated to a particular bit.

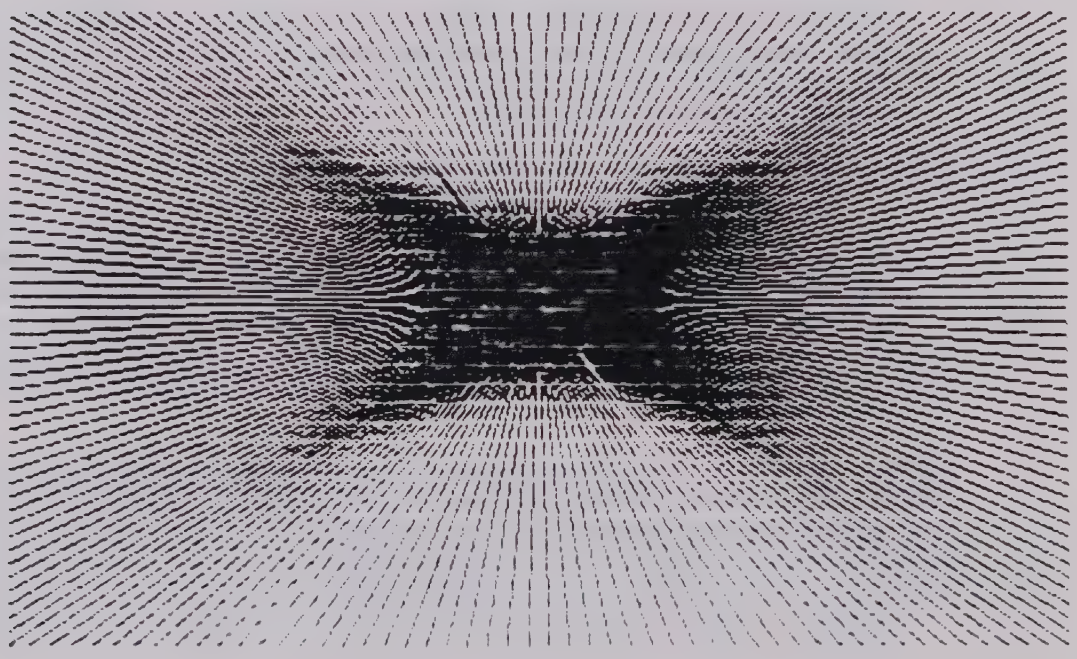
7.11 Display Chip Error Identification

This test is most useful to run when one of the other tests or other visual observations shows errors or instabilities in the display. If the errors are related to a particular memory chip failure on the display card, then this test should identify which chip is at fault. A pattern is written so that the whole display memory contains zeroes except the chip identified on the screen, which will contain all ones. This is done for all 32 memory chips. A bad chip should be able to be detected by careful visual observation of the patterns on the screen.

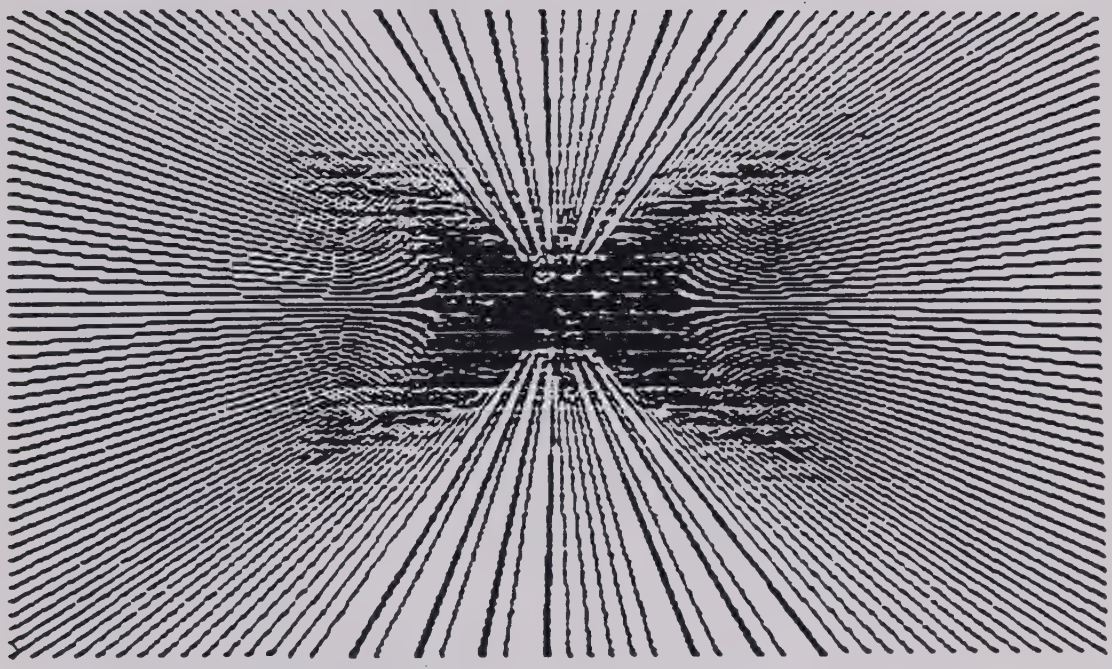
7.12 SUN Test

This is a visual test that shows possible display vector problems not easily recognizable by the display curl test. From the LIVE program, SUN is called by typing SUN (CR).

7.12.1 Typical Good Display



7.12.2 Typical Bad Display



This page intentionally left blank.

Section 8

ADS TESTS

8.0 Purpose

This Section is intended to provide the user with the necessary information required to use the ADS Diagnostic in testing and troubleshooting the ADS unit in a 2511 VCS. The user should be trained on GenRad hardware, and be familiar with conventional troubleshooting techniques and use of standard test equipment.

8.1 Software Requirements

Floppy disk with RT11 operating system and programs shown below.

a. TSD25A
b. ADSD1A
c. ADSD1B
d. ADSAL1

8.2 Hardware Requirements

1. PDP11 Computer (04, 34, LSI-11)
2. 28K Memory (Core or MOS)
3. 2501 Micro CPU
4. Floppy disk
5. Multichannel ADS
6. Sine Wave Oscillator - GR Model 1309 or equivalent

8.3 Reference Material

1. Multichannel ADS Documentation package
2. 2511 System Operating Manual

8.4 Floppy Disk Version

8.4.1 Load a floppy disk with ADS Diagnostics into Unit 0.

8.4.2 Boot RT11 operating program using standard procedures.

8.4.3 Type Load 'ADSD1A' (CR) or 'ADSD1B' (CR)

8.5 General Information

8.5.1 ADS diagnostics have been divided into two sections, ADSD1A and ADSD1B. Type CLEAR (CR) before loading ADSD1A or ADSD1B. They cannot exist in memory simultaneously.

8.5.2 Diagnostics were written using TSL25 and a version of the TSL ADS driver (ADSET). All but three of the routines described here are TSL subroutines and may be listed, edited, and saved as any TSL program. The three routines that are assembly language routines are:

1. RTCTS1 - Real Time Clk Test
1. SDINT1 - Sample Done Interrupt Routine (Must be called by SAMPD)
1. TCTST1 - Transient Capture Test

8.5.3 All of the routines to be described later may be started in the same manner. Type in the name of the program to be run followed by a carriage return. Various programs require different setups. Therefore, it is necessary to describe each routine in detail as to how to set up, operate and how to interpret the results.

8.5.4 An effort will also be made to provide "hints" on how to approach certain problems based on results obtained from each test.

8.5.5 All routines that require a channel to be specified will accept:
'A', 'B', 'C', 'D', or '01', '02', '03', '04'

8.6 ADSD1A Tests

NOTE

The following is a list of test programs included in ADSD1A and ADSD1B in the order that they should be run.

Test#	Name	Description
1.	ADSRW	ADS Read Write Test
2.	RTCTS1	Real Time Clk Test
3.	FDTST	Frame Done Test
4.	SAMPD	Sample Done Interrupt Test
5.	CHTST	Channel X Test
6.	CDTST	Continue/Done Test
7a.	TRIGA	Trigger CH A Test
7b.	TRIGB	Trigger CH B Test
7c.	TRIGC	Trigger CH C Test
8.	TRIGE	Trigger External Test
9.	DIST	Harmonic Distortion Test

8.7 ADSD1B Tests

1.	FSTST	Frame Size Test
2.	SRTST	Sample Rate Test
3.	TCTST1	Transient Capture Test
4.	DIGTRG	Digital Trigger Test
5.	CHSEL	Selector Test, 16 Channel

ADSD1A
(Type CLEAR (CR) before loading ADSD1A)

8.8 ADSRW - ADS Read Write Program

Performs Read Write test on all standard ADS addresses. Determines number of channels and checks all registers.

Tests will print an error message if bad and sequence on to the next register. Test will print ADSRW OK if no errors occur.

8.8.1 Typical Error Message

```
REG R/W Failed
164000                  This printout indicates that bits
Good Data               2^6, 2^7, 2^8 were defective in
777777
Bad Data                register 164000 of ADS
777077
```

8.8.2 Troubleshooting Techniques

To trouble shoot a problem of this nature, it may be desirable to deposit a small program in memory to aid the service engineer. The following is a typical program starting at address $120000_{(8)}$.

```
120000    12737
120002    177777    Move #177777 to register under test
120004    164000

120006    13737
120010    164000    Move contents of register under test to
120012    122000    address 122000

120013    137       Jump back to start
120016    120000
```

8.9 RTCTS1 - Real Time Clk Test (Assembly Language)

This program sets all ones in the RTC register 164016, 2501-4713 board sheet #5 and then waits for the RTC to interrupt on DONE. If no interrupt occurs, program will print the error. The program rings a bell at the start of the countdown sequence and again at the end of 5 passes. This requires the operator to measure the time between bells. This time should be 20 seconds.

8.9.1 Troubleshooting Techniques

Symptom	Possible Problem Areas & Suggestions
Terminal prints: "NO RTC interrupt received"	1. No 4.096 MHz CLK. 2. Counter chain inoperative 3. Put 2501-4713 on extender boards & investigate with scope
Time between bells did not equal 20 seconds	1. Counter is inoperative. 2. Put 2501-4713 on extender boards and investigate with an oscilloscope while running diagnostic.

8.10 FDTST - Frame Done Test

8.10.1 Tests for frame done bit (2^{15} in register 164000) in both the frequency (4.0 MHz) and time (4.096 MHz) domain and prints an error message if no done is received. If done is received, the results are displayed.

8.10.2 The ADS is set up as follows:

Sample Rate = 2 kHz, Gain = ±2 volts, Frame Size = 128,
Free Run, Filters = All Pass., DMA Address is BO.

8.10.3 To stop program, strike the S key on terminal.

8.10.4 Troubleshooting Techniques

	Sympton	Possible Problem Areas
a.	Terminal Prints: "No Frame Done, Time Domain," and "No Frame Done, Freq. Domain."	1. No + SAMP (2501-4712 bd sheet #4) 2. No -FULLBUF (2501-4712 bd sheet #3) 3. No -DMARQST (2501-4713 bd sheet #3) 4. No =OFFCK (2501-4713 bd sheet #4) 5. No +SMPKLD (2501-4713 bd sheet #2, U62-9)
b.	Terminal Prints: "No Frame Done, Time Domain," but Freq domain displays properly.	1. Bad time domain osc. 4.0 MHz oscillator (2501-4712 bd sheet #4)
c.	Terminal Prints: "No Frame Done, Freq. Domain"	1. Bad freq. domain osc. 4.096 MHz oscillator (2501-4712 bd sheet #4)

8.11 SAMPD - Sample Done Interrupt Test (Assembly Language + TSL)

8.11.1 This test ensures that the ADS will interrupt properly to vector address 270 with Sample Done (REG 164000, bit 2^{15}). SAMPD is written in TSL but calls on an assembly language routine SDINT1, which waits for the ADS Done Interrupt to occur, if no interrupt is received, an error message is printed. If the interrupt is OK, the system prints "Sample Done Interrupt Good." This test assumes that all previous tests are operating properly.

8.11.2 Troubleshooting Techniques

Sympton	Possible Problem Areas/Suggestions
System Prints: "Sample Done Did Not Occur"	1. Interrupt request logic band (2501-4713 sht #5). Put board on extender and check for -BURSREQ on U34-5.
	2. Universal Interface is probably OK if the RTCTS0 test passed satisfactorily. If not, check the logic on the ADS universal interface.

8.12 CHTST 'X' - 'X' Channel Sampled Data Test

8.12.1 This test provides the means for testing the channel amplifiers, sample and holds, and filters of channels 1-4.

8.12.2 CHTST can be started 3 different ways:

a. CHTST No parameters. Program determines how many consecutive channels are in system and set up to run those channels.

b. CHTST 'XX' XX = channel to be tested. Program tests only channel requested.

c. CHTST x x = number of channels to be tested, x must be even number.

8.12.3 Performs A/D overload test for channel under test and prints if error occurs (input level must exceed .125 V. to pass test.)

8.12.4 Performs OFFSET adjustment on all channels initially and with every change to filter setting, gain setting, or AC/DC coupling, and rings bell if error and prints OFFSET ERROR on CH# (CH), for channels that are bad.

8.12.5 Requires input to channel(s) of approx. 30 Hz at 4 volts P-P.

8.12.6 Strike G key to change gain. In response to ?, type in actual voltage range to be used, i.e., ? .125(CR) = ±.125 volts. Gain settings:

0.125 volts	2.0 volts
0.25 volts	4.0 volts
0.5 volts	8.0 volts
1.0 volts	16.0 volts

8.12.7 Strike A to perform automatic offset test on all selected channels, stepping through all voltages. Program will print channels that fail at any particular voltage. Routine returns to 2 volt scale when complete.

8.12.8 Strike F key to change filter setting. In response to ?, type in the filter range desired, i.e., ? 1000 (CR) = 1000 Hz. The following is a table of cutoff frequencies to be tested. Increase the input frequency above the cutoff frequency selected and observe the attenuation of the signal.

Frequency
10 Hz
250 Hz
1000 Hz
2000 Hz
2500 Hz
4000 Hz
8000 Hz
10000 Hz
25000 Hz
0 = All Pass

8.12.9 Strike C key to change coupling. In response to ?, type in 0(CR) for DC, and 1(CR) for AC.

8.12.10 ADS is initially set up as follows.

Sample Rate = 2 kHz, Gain = ±2 volts, Frame Size = 128, Free Run, Freq cutoffs = 800 Hz.

8.12.11 This test can be used to check cutoff frequency of all channels at the same time.

8.12.12 Strike S key to stop.

8.12.13 Troubleshooting Techniques

Symptom	Possible Problem Areas & Suggestions
Displayed Data is missing information or totally incorrect	1. Try same test on the other channel, if it is good, then problem is in the 1923-4981 board under test. Put the board on the extender and proceed troubleshooting. 2. If problem is on all channels, check: a. The input to the A/D on 2501-4711 board. b. Look at all 12 bits out of A/D & verify they are all toggling. c. Look at output of FIFO on the 2501-4713 board, sheet #3.
Gain, filter or AC coupling does not function properly.	3. Problem is probably in the 1923-4981 board of the channel under test. Put the board on an extender and proceed with troubleshooting.
System prints "OFFSET ERROR"	4. System may need realignment, or there is excessive offset on a given filter or gain setting caused by defective component.

8.13 CDTST - Continue Done Test (Double Buffered Results)

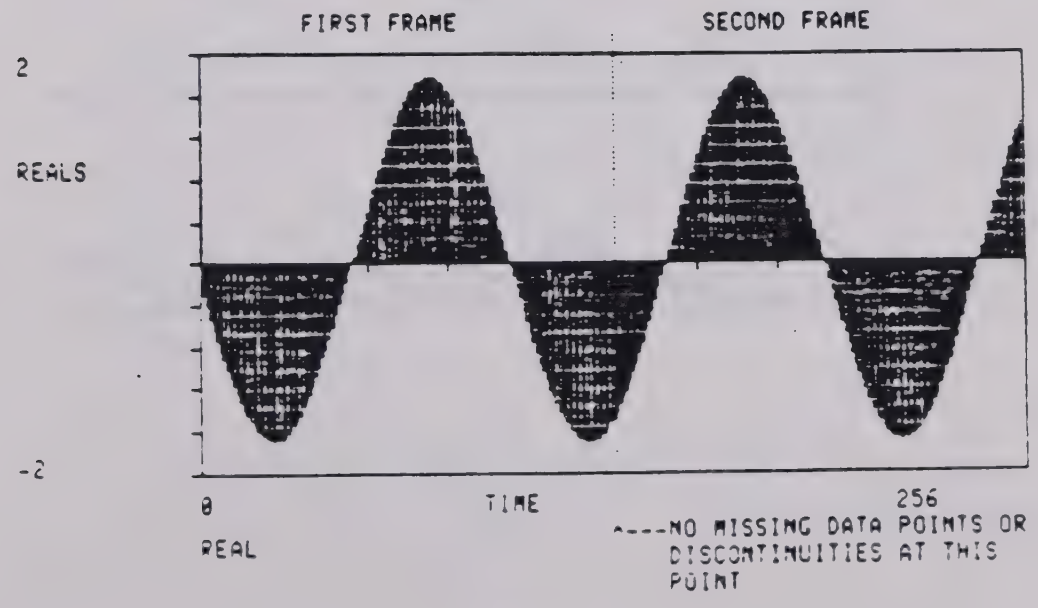
8.13.1 This test checks out the continuous operation mode of the ADS. Basically the test sets up the ADS for sampling data in continuous, sets sample enable, and waits for frame done. Frame done is cleared and the test waits for the second frame done and then displays the two completed frames of data. Any problem that would occur would be in the center of the data frame where the "continued" data begins. A vector on the center line of the display shows the frame boundaries.

8.13.2 Input should be 10 Hz 4 volts P-P.

8.13.3 Troubleshooting Techniques

Symptom	Possible Problem Areas & Suggestions
Missing data points or discontinuities in the data frame.	1. Continue-Done circuitry on 2501-4713 board sheet #2 is not operating properly. Modify CDTST to skip the display and continuously run the program. Put the 2501-4713 board on an extender and verify proper operation. 2. Alternate frame buffer logic 2501-4712 board sheet #3 is not operating properly. U18-8 should toggle at the end of each continued data frame. Also, the offset codes DS0-DS3 going to address logic on 2501-4712 board sheet #2 may be inoperative.

8.13.4 Typical Display



8.14 TRIGA, TRIGB, TRIGC - Channel X Trigger Test

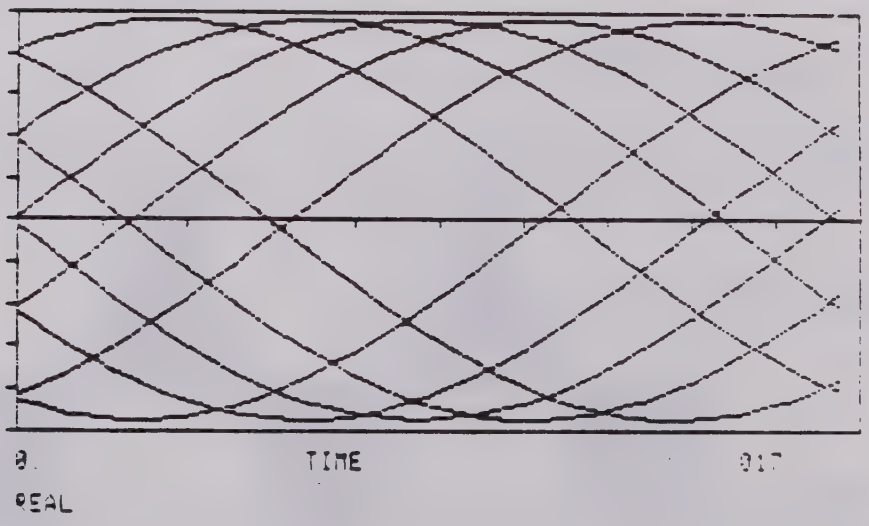
8.14.1 These tests check the trigger level on channel A, B and C respectively. They require a 4 volt P-P, 30 Hz signal into the channel being tested, and requires the operator to visually check for the proper trigger levels on the display. It is necessary to have aligned the zero on the trigger board (1923-4982) prior to running this test. Basically the program triggers positive and negative on -80%, -40%, 0%, +40%, +80% of the full scale input to the trigger board. The operator must verify that the trigger is within 5% of the previously mentioned levels.

Note: The input must be very close to 4 volts P-P to obtain the proper values.

8.14.2 Troubleshooting Techniques

Symptoms	Possible Problem Areas & Suggestions
No trigger on any channel or trigger levels are incorrect.	1. Problem is probably on trigger boar 1923-4982. Put the board on an extender and run TRGCAL to troubleshoot no trigger problem. It may be convenient to create a special test program if trigger levels are incorrect.

Typical TRIGA Display



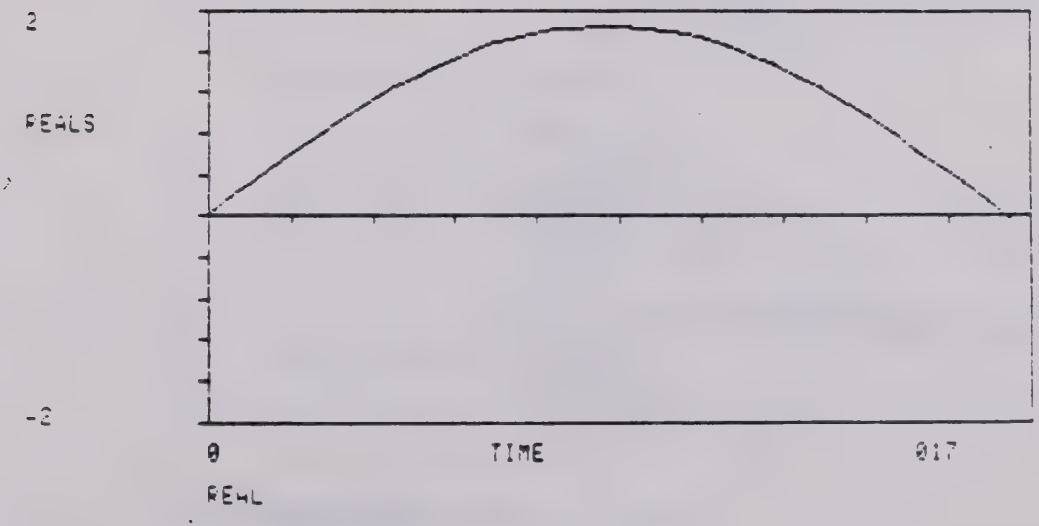
8.15 TRIGE - External Trigger Test

8.15.1 This test checks the external trigger input to the ADS. It requires an input to channel A and external trigger at 4 volts P-P at 30 Hz. The trigger level on external trigger is set to 0% of full scale only (Positive slope) and simply verifies that external triggering is operative. Remember, external triggering is 4 volts P-P input full scale.

8.15.2 Troubleshooting Techniques

Symptoms	Possible Problem Areas and Suggestions
No trigger in external.	1. Problem is selector switch on 1923-4982 board or the cable from EXT TRIG on rear of ADS to the trigger board. The rest of the circuitry is used for TRIGA, TRIGB, TRIGC.
No display on Channel A.	1. Channel A not connected.

Typical Display of TRIGE



8.16 DIST - Harmonic Distortion Test

8.16.1 This test displays the PSD on each channel to show harmonic distortion. The test determines how many channels (must be consecutive) the system has, and steps through each channel displaying first the input signal to check for over-range and then the power spectrum. The program pauses after each PSD.

8.16.2 Input should be a 12 kHz 2V P-P sine wave.

8.16.3 The 24 kHz peak (second harmonic) should be below 60 dB.

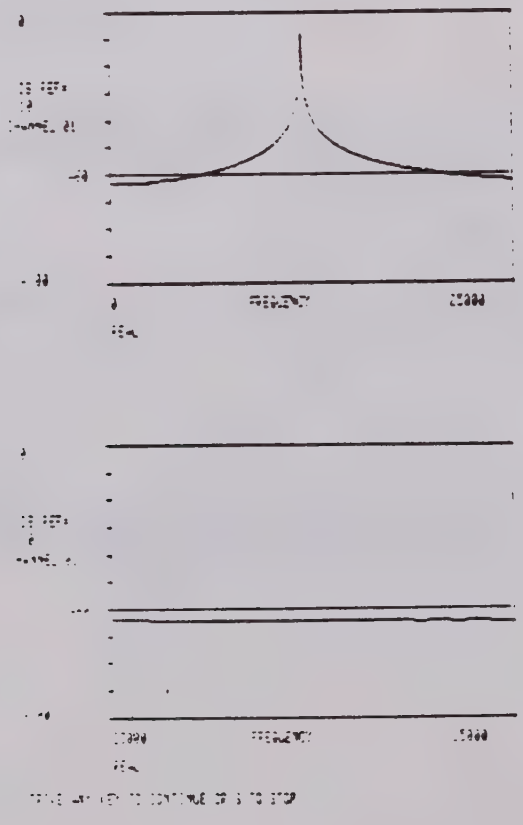
8.16.4 Trouble Shooting

a. An over-range input signal will cause a series of harmonic spikes - keep input below 2V P-P.

b. Most distortion problems are caused by 1923-4981 filter board.

c. An individual channel can be run by typing:
ADPSD 'xx'. ('xx' = channel #01, 02. . .08)

8.16.5 Typical Display of DIST.



ADSD1B

(Type CLEAR (CR) before loading ADSD1B)

8.17 FSTST - Frame Size Test

8.17.1 This test checks the various frame sizes from 128-8192 by setting the sample rate to 2 KHz and having the operator verify that the number of cycles displayed is approximately equal to the number printed. Frame size of 16,284 are not checked due to memory limitations with this program.

8.17.2 Strike S key to stop program.

8.17.3 Strike any other key to proceed to next test.

8.17.4 Requires input of 16 Hz at 4 volts P-P.

8.17.5 Troubleshooting Techniques

Symptoms	Possible Problem Areas & Suggestions
Number of cycles displayed does not match the number printed.	1. Oscillator not set to proper frequency. 2. Problem in addressing logic (2501-4712 board sheet #2, #3).
Displayed data is totally incorrect on one or several different frame sizes	1. Addressing logic problem (2501-4712 board Sheet #2, #3). 2. Put 2501-4712 board on extender and using TSL, create the following test program to aid in troubleshooting.

```
***
Create Test
10 HOLIN I1
20 BITM5, 128, I1
30 IF I1, 83, 40, 60, 40
40 FSTST XXX, 2097 (XXX is frame to be tested)
50 GOTO 10 (Prog. will simply loop & constantly input data to IN the frame size specified)
60 Return
EDIT FSTEST
DELETE 80
```

8.18 SRTST - Sample Rate Test

8.18.1 This program performs the following tests. In the description, SM refers to Sample Multiplier bits 2^{12}, 2^{13} in register 164004 and N refers to the count, bits $2^0 - 2^{11}$ in register 164004. Together the SM and N determine the sampling frequencies to be used, and the three following tests should check all hardware combinations.

8.18.2 Test #1, Sample Rate = 1 KHz, SM = 00, N = 0

8.18.3 Test #2, Sample Rate = 100 Hz, SM = 01, N = 0

8.18.4 Test #3, Sample Rate = 10 KHz, SM = 10, N = 4094

8.18.5 Program automatically tests the oversample bit in Register 164004 (bit 2^{15}).

8.18.6 These tests require the operator to set the oscillator to the frequency specified on the terminal, and to verify that the FFT results peak at 200 Hz, 20 Hz and 2 KHz respectively.

8.18.7 Troubleshooting Techniques

Symptoms	Possible Problem Areas & Suggestions
System prints: "Error OVRSMP bit set in REG 164004" -OR- "Error, OVRSMP bit not cleared by sample enable"	1. OVRSMP logic defective (2501-4713 board Sheet #2) (U31-9). 2. Sample rate generator not working properly (2501-4712 Sheet #4).
Displayed results not at proper frequency.	1. Sample rate generator defective (2501-4712 board, Sheet #4).

8.19 TCTST1 - Transient Capture Test (Assembly Language)

8.19.1 This program expects the remainder of the ADS to be working properly. The program is testing two basic conditions.

8.19.2 Transient capture delay (164006) set to 3777. Sample rate at 2 KHz. Tests for sample done interrupt occurring within 2048 ms after counter starts.

8.19.3 Transient capture delay (164000) set to 3777, sample rate at 2 KHz, tests that sample done interrupt occurs at least 2045 ms after counter starts.

8.19.4 Requires input to channel A at 1 KHz at 5 volts P-P.

8.19.5 Test prints out on error and on test complete.

8.19.6 Troubleshooting Techniques

Symptoms	Possible Problem Areas & Suggestions
Program never completes.	1. No analog input to channel A, or trigger circuit is defective.

Troubleshooting Techniques (cont'd)

Symptoms	Possible Problem Areas & Suggestions
System prints: "Trans Cap Req'd more than 2047 counts to generate TRANSTP"	1. Transient capture counter defective. (2501-4712 Sheet #3)
-OR-	
"Trans Cap Req'd less than 2047 counts to generate - TRANSTP"	

8.20 DIGTRG - Digital Trigger Test

8.20.1 This program sets the ADS up for digital triggering and waits until a ground potential is felt at the digital trigger input at the rear of the ADS box. When the trigger is received, the data is taken through channel A and displayed on the CRT.

8.20.2 Troubleshooting Techniques

Symptoms	Possible Problem Areas & Suggestions
No display, system hangs up waiting for trigger.	1. No ground available at digital trigger BNC.
	2. Digital Trigger logic defective. (2501-4713 Sheet #2)

8.21 CHSEL - Selector Test (Option)

8.21.1 This program allows the operator to check out the operation of the 16 channel selector option to the ADS. The test displays a 4 volt P-P signal from the selected channel. Initially channel = 1.

8.21.2 Strike C key to change to new channel. After ?, type the desired channel numbers as they appear on the selector. I.e., numbers 1— 16.

8.21.3 Operator must insure output of selector is connected to Channel A of the ADS, and that an analog input is connected to the channel under test.

Channel being tested is displayed on monitor.

8.21.4 Strike S key to stop program.

8.21.5 Troubleshooting Techniques

Symptoms	Possible Problem Areas and Suggestions
No Display	1. Defective selector assembly.
Display Data on other then selected channel.	1. Defective QDAS PC3 (2511-4701). 2. Defective selector assembly. 3. Defective cables.

Section 9

RT11 STANDALONE TESTS

9.1 Phase Matching Tests (Internal use only)

9.1.0 Test Equipment Required

Requires a Random noise source 1 V p-p.

9.1.1 Description of Test

Phase is a test that displays the phase matching of the 1923-4981 filter boards, one against the other. The test determines how many channels are in the system and then computes a transfer function of one channel against another. All channel combinations are run, as are the 3 frequency ranges (25 kHz, 2.5 kHz, 250 Hz). The channels gain are run at 2 volts and 1 volt level. The input should be random noise ± 1 volt peak.

9.1.2 Loading Instructions

Run TSD25A and type Load 'PHASE' (CR). Type PHASE (CR) to start program or type PHASE X CR only X band width is checked.
X = bandwidth.

NOTE: Type 'CLEAR' before loading program if other tests have previously been run.

9.1.3 Input

Input is Random noise ± 1 volt.

9.1.4 Display

The display shows the phase relationship between the two indicated channels. Range is 5^{o}. The maximum difference should be 5 degrees. To change scales, type SC (CR), then the scale you want, XX(CR), then G(CR).

To continue to the next channel, type R(CR).

9.1.5 Trouble Shooting

Phase matching is almost always a function of the filter boards (1923-4981) and is usually a problem with the precision capacitors. See examples of typical displays.

9.1.6 Typical Displays

9.2 MEMTST

9.2.0 Memory Test Description

This test is used to test the extended memory from 28-124 K on the LSI-11/2.

Since only the 2501 microprocessor can access this area of memory for data storage and manipulation, the 2501 processor must be used to test this area of memory.

This test is somewhat slow because it is written in TSL.

9.2.1 Loading Instructions

Run TSD25A and type Load 'MEMTST'(CR). To start program type MEMTST, Y, Z or MEMTST, PAGE, X, Y, Z where:

X = Page number (0-3, depending on amount of memory on system)
page = 32K

If only one page of the memory is to be tested, then the page can be specified by the starting option:

MEMTST PAGE X = Page # 0-3

Only one page can be run with this option.

Y = Test number (±1 to ±4)

A minus sign (-) before the integer will suppress the test number printouts on the terminal.

Z = Number of errors printed (0 = special case)

A second parameter can be used to limit the number of errors that are logged on the printer. (Default = 30) If there is no printer on the system.

A zero is entered for Z option. The system will halt on the first memory error.

If the printer is turned off, this parameter should be set to either 0 or a minus number or the test may hang.

9.2.2 Options

There are several options with this program, called by using an integer number after the name. The parameters are as follows:

Test Number

MEMTST 1: Short test - one pass

MEMTST 2: Short test - continuous

MEMTST 3: Long test - one pass

MEMTST 4: Long test - continuous

Test Termination:

Test can be terminated at any time by pressing the 'STOP' Key. The program will halt after completing the current block. This stop will give an error message of error 45.

9.2.3 Error Messages

Error addresses and data are given in decimal notation. The program tests 4K blocks one at a time, starting at 28K-32K.

The error messages give the page number of the error.

This is the memory board. Page 0 = first board, Page 1 = second board (in ascending address order).

NOTE: If printer is not on system the program will halt when an error is detected after the error is displayed on CRT.

Because of a limitation of TSL/TSD, if the program attempts to print the number 32768 as an error address or data, the program will halt with an error 33.

9.3 QDAS Tests

This section provides the user with the necessary information to run the QDAS diagnostics on the 2511 VCS. The user should be trained on GenRad hardware and be familiar with conventional troubleshooting techniques and use of standard test equipment.

9.3.1 Software Requirements

A. 5¼" flopply with RT-11 operating system and the following programs are required. (2511-0691 diskette)

1. TSD25A.SAV
2. QDASA.SAV
3. QDASB.SAV
4. QDASC.SAV
5. QDAS1.TSL
6. QDAS2.TSL

9.3.2 Hardware Requirements

A. 2511 VCS unit consisting of:

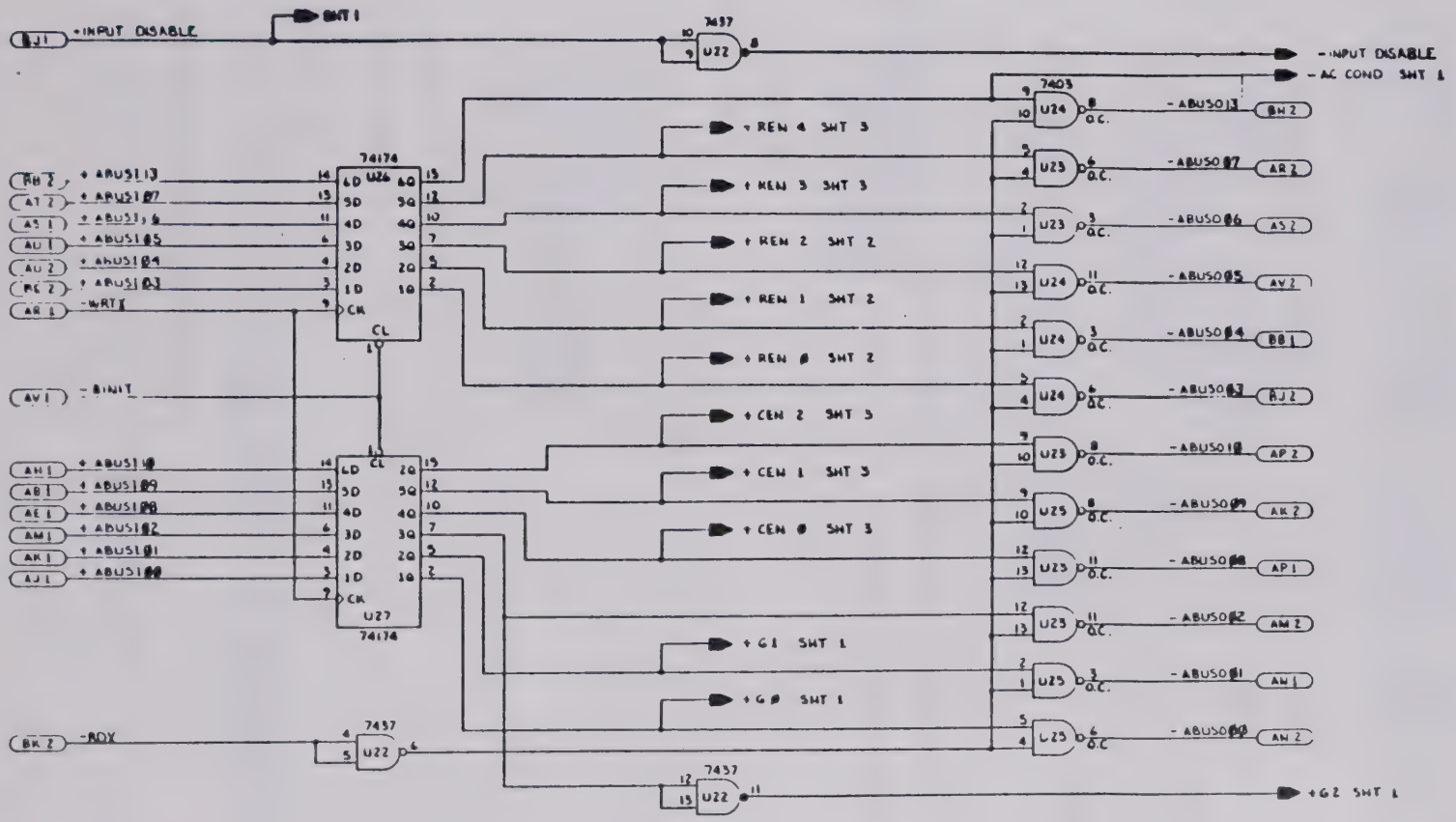
1. LSI-11 cpu
2. 32Kw memory
3. 2501 uCPU
4. ADS unit
5. 5¼" floppy disk
6. CRT
7. Keyboard
8. QDAS PCB

9.3.3 Test Equipment Requirments

1. Dual trace scope
2. Digital volt meter
3. Sine wave generator
4. Voltage source

9.3.4 General Information

The ADS portion of the 2511 VCS must be calibrated and tested as defined in Section 10. The QDAS tests utilize the ADS input for displays and DFT calculation. An uncalibrated or defective ADS will cause erroneous results.



TIME/DATA

SCHEMATIC – ANALOG INPUT AND FILTER

D 1923-4981-21 13

9.3.5 Loading and Testing Procedures

Abstract

The QDAS tests are separated into two sections. QDAS1 and QDAS2 are subroutines of TSL and are loaded via TSD25A diagnostic TSL. QDASA, QDASB and QDASC are assembly language tests and run as .SAV programs from RT-9.

Test Loading - TSL Subroutines

1. Load the RT-11 operating system using standard procedures.
2. Run TSD25A by typing R TSD25A (CR)
3. At the next '>' prompt type LOAD 'QDAS1' (CR) or LOAD 'QDAS2' (CR). (Between loading QDAS1 and QDAS2 type CLEAR (CR) to prevent error messages of duplicate subroutine loads.)

   Once the TSL subroutine is loaded the '>' prompt is displayed. See paragraph 9.7 and 9.12 for individual testing.

Test Loading - Assembly language tests

1. Load the RT-11 operating system using standard procedures.
2. Type R QDASA (CR)
3. Control C (^C) will return to the '.' prompt of RT-11 for loading of QDASB or QDASC.

9.3.6 Test Execution

Sequence

QDAS tests should be run in the following order.

1. QDAS1 (TSL)
2. QDASA (Assembly Language)
3. QDASB (Assembly Language)
4. QDASC (Assembly Language)
5. QDAS2 (TSL)

External connections

For all tests conect DRIVE to ADS channel 1.

9.3.7 QDAS1 (TSL Subroutine)

All test subroutines in QDAS1 are executed from the TSL prompt '>' by typing the subroutine name followed by a carriage return (CR).

Example: > subroutine name (CR)

All tests will run in sequence by typing MSTRLP (CR).

9.3.8 REGTRP - QDAS Register Test

Function

Verifies all 8 QDAS registers are present. An error will cause the RT-11 operating system to time out trap (trap to 4).

Troubleshooting Techniques

Using ODT in the LSI-11, open and read the eight registers to determine the defective register. Only the first register is read/write.

Test Messages

```
>REGTRP (CR)
IF THIS TEST TRAPS TO 4 IN THE MONITOR
  THEN THE Q-BUS INTERFACE OF THE Q-DAS IS
    SUSPECT.
DID REGISTER 0
DID REGISTER 1
DID REGISTER 2
DID REGISTER 3
DID REGISTER 4
DID REGISTER 5
DID REGISTER 6
DID REGISTER 7
TEST COMPLETE, NO FAILURES.
```

9.3.9 REGBIT - QDAS CSR Bit Set/Clear Test

Function

Verifies the bits in the CSR can be set and cleaned. Failing bits will be displayed on the CRT.

Troubleshooting Techniques

Use the LSI ODT program to manually test the failing bit.

Test Messages

```
>REGBIT (CR)

TESTING BITS IN CSR
NO BIT FAILURES IN CSR REGISTER.
```

9.3.10 FLCONT - Filter Continuity Test (I15=O)

Function

Test checks for filter continuity by passing +_ full scale voltage through the filter. The program will flag a filter that is out of tolerance. With I15=O no zero offset is performed.

Test Messages

Typical printout.

```
>FLCONT

TESTING FOR FILTER CONTINUITY
FILTER  10U      OU       -10U              ADJ          ERR  ÷FS
-------------------------------------------------------------------
0        9.973   -.0378   -10.03   10.01    0  -10.       .0585
1       10.      -.0115   -10.02   10.      0  -10.01     .0781
2       10.      -.0078   -10.     10.      0  -10.      0.
3        9.962   -.0390   -10.03   10.      0  -10.       .0024
4       10.       .0078   -10.     10.      0  -10.      0.
5        9.977   -.0234   -10.02   10.      0  -10.       .0024
6       10.      -.0078   -10.     10.      0  -10.       .0045
```

9.3.11 FLCONT - Filter Continuity with Zero Offset (I15=1)

Function

This test uses the zero offset circuitry to zero the output of the QDAS. Errors will be flagged.

Test Messages

Typical printout.

```
>LET I15,1
>FLCONT TESTING ZERO OFFSET FUNCTION.
                    OUTPUT IS IN MILLIVOLTS
 FILTER  LOOP BIAS  OFFSET V.  MEAS. ZERO  ERR % FS
-------------------------------------------------------
    0      -59.04     54.41      -4.631      .04631
    1      -41.8      37.02      -4.787      .04787
    2      -36.08     31.03      -5.042      .05042
    3      -64.49     60.34      -4.15       .0415
    4      -18.7      15.5       -3.205      .03205
    5      -47.48     40.96      -6.515      .06515
    6      -30.03     25.15      -4.875      .04874
```

Test Messages (continued)

Example of error printout. Percent of error shield be less than .1%.

TESTING ZERO OFFSET FUNCTION.

OUTPUT IS IN MILLIVOLTS

FILTER	LOOP BIAS	OFFSET V.	MEAS. ZERO	ERR % FS
0	-58.94	54.44	-4.507	.04507
1	-41.77	36.95	-4.824	.04824
2	-36.13	30.93	-5.2	.052
3	-64.41	60.28	-4.131	.04131
4	-18.7	13.55	-5.157	.05157
5	-47.48	0.	-47.48	.4748
6	-30.03	25.13	-4.904	.09404

FILTER 5 OUT OF TOLERANCE

9.3.12 QRAMP - Output Ramp

Function

This test load a ramp (4096 values from -10V to +10V) into memory and dumps the ramp into the QDAS via the QDAS DMA channel. Any discontinuities indicate a failure in the DMA section of the QDAS.

Test Connection

Connect scope to the drive out BNC.

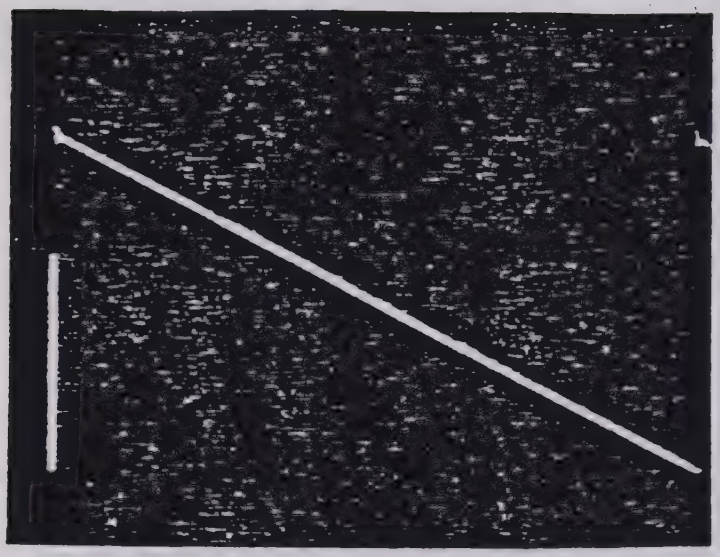
Set the scope as follows:

Vertical - 5 volt/div
Horizonal - 20 ms/div
Trigger - normal, internal

Test Messages

>QRAMP
LOADING THE OUTPUT RAMP INTO MEMORY
LOADING COMPLETE

Typical Display



9.3.13 MUXREG - 16 Channel Scanner (Option)

Function

The test toggles the 4 selector bits for the 16 channel scanner option.

Test Connection

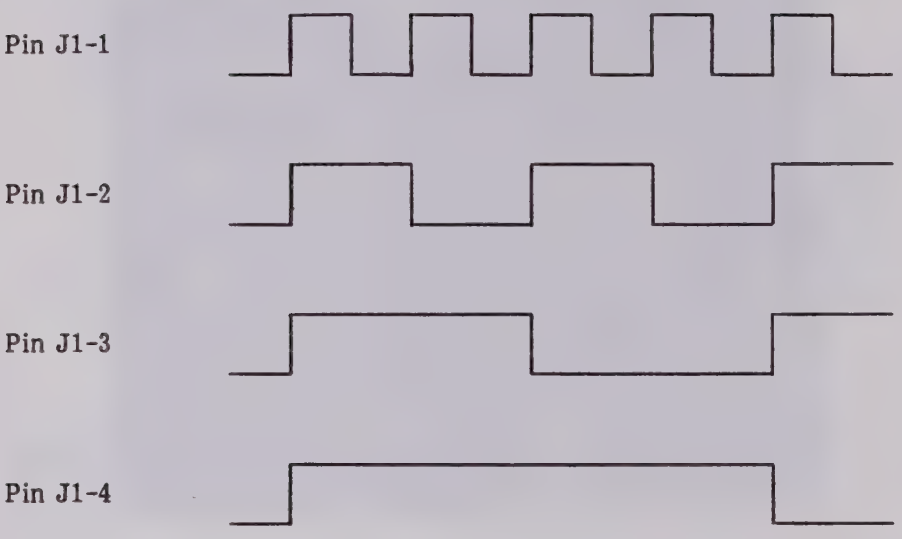
While the test is running a scope is used to check the bit toggles at the 16 channel connector on the rear of the 259.

Test Messages

```
>

MUXREG
OUTPUTTING THE BINARY COUNTING SEQUENCE TO
THE MULTIPLEXING REGISTER
COUNTING SEQUENCE DONE
TO RERUN THIS TEST TYPE MUXREG (RETURN)
```

Typical Display



9.3.14 MSTRLP - Master Loop

This test will run all of the preceeding tests in the order presented. At the TSL prompt '>' type MSTRLP (CR)

9.3.15 QDASA (Assembly Language)

Function

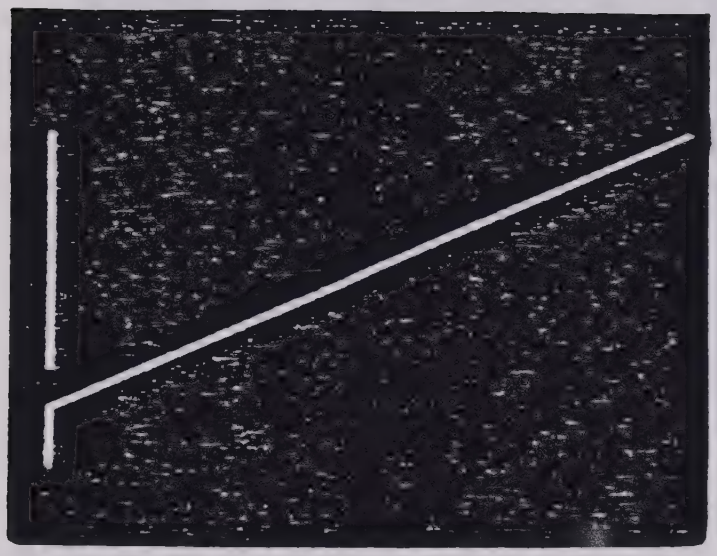
This test is similar to QRAMP except that the buffer is broken into 16 segments and then spliced together with interrupt processing to form the same output ramp as in QRAMP. Discontinuities in the ramp indicate a problem in the interrupt logic on the QDAS.

Test Connection

Connect scope to the Drive BNC. Set the scope as follows:

Vertical - 5 volt/div
Horizontal - 20 ms/div
Trigger - normal, internal.

Typical display



9.3.16 QDASB (Assembly Language)

Function

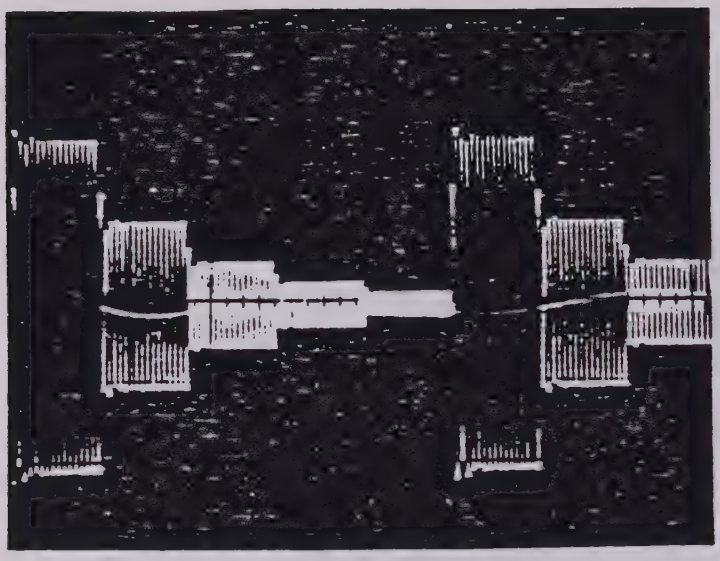
Outputs a 5120 Hz sine wave that changes amplitude by half 4 times, returns to full scale and repeats. Deviation from the typical display indicates a problem with the MDAC.

Test Connection

Connect a scope to the Drive BNC. Set scope as follows:

Vertical	-	5 volt/div
Horizontal	-	2 ms/div
Trigger	-	Normal, internal

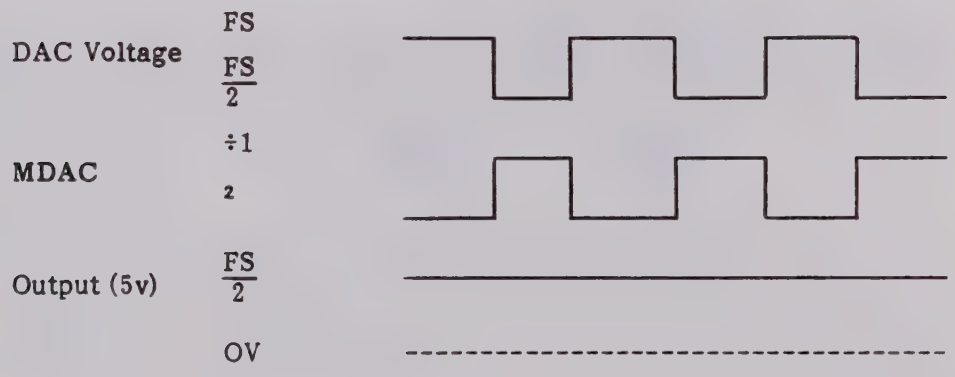
Typical Display



9.3.17 QDASC (Assembly Language)

Function

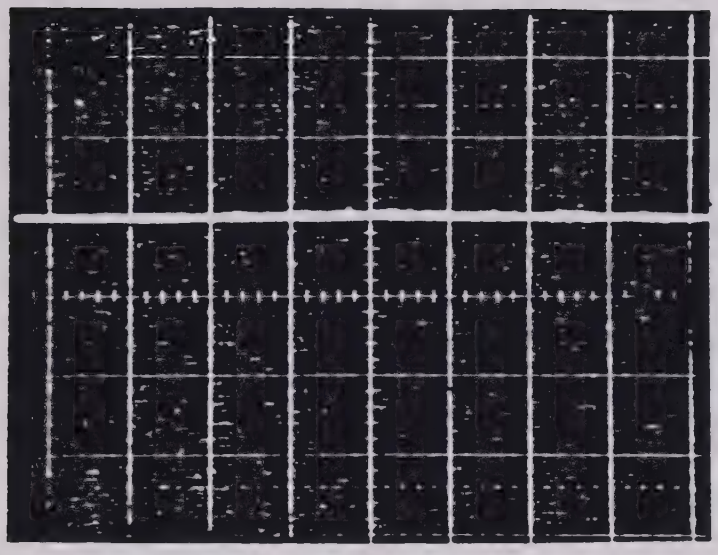
Tests the ability of the MDAC to load correctly at End of Frame (EOF). The test outputs two buffers. Buffer #1 is a D.C. level representing full scale (FS). Buffer #2 is a D.C. level of FS divided by 2. The two buffers are continuously output one after the other. During the output of Buffer #1 (Full Scale) the MDAC is loaded with a divide by 2 to produce $\frac{\text{Full Scale}}{2}$. During the output of Buffer #2 $\frac{(\text{Full Scale})}{2}$ the MDAC is loaded with divide by 1. The output at the Drive BNC should remain at $\frac{\text{Full Scale}}{2}$.



Test Connection

Connect scope to Drive BNC.

Typical Display



QDAS2 (TSL)

All test subroutines in QDAS2 are executed from the TSL prompt '>' by typing the subroutine name followed by a carriage return (CR).

All tests wil run in sequence by typing MSTRLP (CR).

9.3.18 FILATN - Filter Attenuation

Function

The test will check each filter for bandpass ripple and out of bond rejection. Errors are displayed on CRT.

Test Connection

Connect BNC cable from drive out to ADS channel A.

NOTE

The ADS channel A must be properly aligned. Occasionally FILATN will fail due to the Q-DAS test specs being higher than the ADS input specs. If replacing the Q-DAS PCB does not alter the error, proceed by replacing or recalibrating the ADS input channel A.

9.3.19 SWITCH - Switching Transient

Function

The test creates a worst care switching transient response for the zero crossing MDAC load mode, the program loads a 20Hz sine wave and outputs the buffer via DMA operation. The amplitude is switched at each zero crossing.

Test Connection

Connect Scope to Drive BNC. Fluctuation in the sine wave is normal. Set scope to:

Vertical	-	50 mv/div
Horizontal	-	.5 ms/div
Trigger	-	DC, Auto, internal

Test Messages

```
>SWITCH
TO RUN THIS TEST AGAIN TYPE FLOP (RETURN).
>FLOP
>FLOP
```

Typical Display

Crossings must be less than 8 mv of transient noise.

Test Messages

```
>FILATN

QDAS DIAGNOSTICS REV 1.0
FILTER # 0 THE 200 HZ FILTER

ESCAPE TO ABORT — HIT RETURN TO CONTINUE
```

Typical Printout

```
AT  800 HZ. ATTENUATION IS 0. DB. LIMIT (+-) .2
AT 1600 HZ. ATTENUATION IS -.002441 DB. LIMIT (+-) .2
AT 2400 HZ. ATTENUATION IS .001525 DB. LIMIT (+-) .2
AT 3200 HZ. ATTENUATION IS -.004578 DB. LIMIT (+-) .2
AT 4000 HZ. ATTENUATION IS -.0299 DB. LIMIT (+-) .2
AT 4800 HZ. ATTENUATION IS -.047 DB. LIMIT (+-) .2
AT 5600 HZ. ATTENUATION IS - .03204 DB. LIMIT (+-) .2
AT 6400 HZ. ATTENUATION IS -.02777 DB. LIMIT (+-) .2
AT 7200 HZ. ATTENUATION IS .004578 DB. LIMIT (+-) .25
AT 8000 HZ. ATTENUATION IS .122 DB. LIMIT (=-) .4
DO YOU WISH TO RUN THE ANTI ALIAS CHECK?
Y OR N
```

Example of error printout.

Less than .2 DB for 100 through 800 Hz.
Less than .25 DB for 900 HZ.
Less than .4 DB for 1000 Hz.

Test limits.

```
AT  100 HZ. ATTENUATION IS 0. DB. LIMIT (+-) .2
AT  200 HZ. ATTENUATION IS -.02868 DB. LIMIT (+-) .2
AT  300 HZ. ATTENUATION IS -.07568 DB. LIMIT (+-) .2
AT  400 HZ. ATTENUATION IS -.158 DB. LIMIT (+-) .2
AT  500 HZ. ATTENUATION IS -.2777 DB. LIMIT (+-) .2 <-ERROR
AT  600 HZ. ATTENUATION IS -.4132 DB. LIMIT (+-) .2 <-ERROR
AT  700 HZ. ATTENUATION IS -.5521 DB. LIMIT (+-) .2 <-ERROR
AT  800 HZ. ATTENUATION IS -.7376 DB. LIMIT (+-) .2 <-ERROR
AT  900 HZ. ATTENUATION IS -.954 DB. LIMIT (+-) .25 <-ERROR
AT 1000 HZ. ATTENUATION IS -.211 DB. LIMIT (+-) .4 <-ERROR
DO YOU WISH TO RUN THE ANTI ALIAS CHECK?
Y OR N
FILTER # 3 THE 2000 HZ FILTER
```

This page intentionally left blank.

Section 10

XXDP+ GUIDELINES

10.0 XXDP+ Naming Conventions

In deriving the naming conventions for XXDP+, an attempt has been made to clear up some of the naming inconsistencies in the XXDP software which XXDP+ replaces.

Note: In the following descriptions, the special symbols used have the indicated meaning:

dm Device mnemonic (DK = RK11/RK05)

rp Revision level (r), Patch level (p)

yy Two arbitrarily chosen characters

It should also be noted that .SYS files are binary files which may be loaded, but cannot be directly executed.

The following rules govern all XXDP+ product related names:

1. Device mnemonics are identical to those used within the Digital PDP-11 software community unless it is impossible to comply.

2. All XXDP+ software has a Diagnostic Package Code starting with the letters "CH".

3. XXDP+ Monitors:

   a. Package Code/Title format id:

      CHPdmIP XXDP+ dm MONITOR BIN

   b. Filename format of binary file is:

      HMdmrp.SYS

   c. Filename format of core image file (on sequential access mass storage devices only) is:

      HMdmrp.SAV

4. XXDP+ Device Drivers:

   a. Package Code/Title format is:

      CHDdmrp XXDP+ dm X DRV BIN

      (Note: x is: "R/W" for Read/Write
      "R/O" for Ready Only
      "W/O" for Write Only

   b. Filename format of binary file is:

      HDdmrp.SYS

5. XXDP+ Utility Programs:

   a. Package Code/Title format is:

      CHUyyrp SSDP+ etc.

   b. Filename conventions are random dependent on human interfacing requirements. They may or may not be derived from the package code.

6. All PDP-11 Diagnostic Supervisor Versions:

   a. Package Code/Title format is:

      CHSyyrp etc.

   b. Filename format of binary file is:

      HSyyrp.SYS

7. XXDP+/PDP-11 Diagnostic Supervisor Miscellaneous:

   a. Package Code/Title format is:

      CHOyyrp etc.

   b. Filename conventions, if files are present, are random dependent on human interfacing requirements. They may or may not be derived from the package code.

8. XXDP+ Package Names will be derived from the device mnemonic as follows:

   dmDP+

   For example, the XXDP RK05 Package, RKDP, becomes the XXDP+ package DKDP+.

10.1 XXDP+ Media Packaging Guidelines

The following guidelines must be followed when building all types of XXDP+ mass-storage media.

1. Every piece of media must contain at a minimum the following files in the order given at the very beginning of the medium.

   a. Bootable monitor(s) for that medium

   b. HSAArp.SYS (CHSAArp XXDP+ DIAG SUPP BIN)

   c. HUDIrp.SYS (CHUDIrp XXDP+ MONTR DIR BIN)

   (The bootable monitors for random access devices do not appear in the directory. The bootable monitors for sequential access devices, magtapes and TU60 Cassettes, appear as the files HMdmrp.SAV files.)

2. For each type of media, where there is sufficient storage capacity, there must be one medium that contains at a minimum the following files in the order given immediately following the files specified in Guideline #1.

   a. HELP.TXT (CHOHLrp XXDP+ HELP TEXT FILE)

   b. HDdmrp.SYS (CHDdmrp all XXDP+ device drivers)

   c. All XXDP+ utility programs (CHU????) not already mentioned.

   d. All versions of the Diagnostic Supervisor (CHS????) not already mentioned.

   e. HHdmrp/SYS (CHMdmrp all XXDP+ monitor binaries)

3. For those media types which do not have enough storage capacity to comply with Guideline #2, the following applies:

   a. Always satisfy Guideline #1.

   b. Accomplish as much of Guideline #2 as possible.

   c. UPD2.BIB (CHUP2rp) must coexist on the same medium with all device drivers (CHD????).

This page intentionally left blank.

Section 11

XXDP+ UPDATE PROGRAM

11.0 Introduction

XXDP+ has one utility program to be used for file maintenance. This program is called UPD2. The UPD2 program will run on an 8K system but can not load a 4K file there.

The UPD2 program is not stand alone and shares memory with the monitor. UPD2 uses certain functions made available by the monitor. The exit command is used to return to the monitor.

UPD2 contains read/write drivers for devices. These drivers are resident on the system device as SYS files. They are transferred to memory as they are needed.

Utility program prints an asterisk (*) as a prompt for command mode. The valid commands for the UPD2 utility are next described. Some of the commands described below allow the use of wildcards in the filename and extension. Wildcards are the * and % characters. These characters may be placed in filenames and extensions to indicate that any characters are valid in the characters positions indicated by the wildcards. The * wildcard indicates all character positions from the position of the * to the end of the filename or extension. The % wildcard indicates only the character position occupied by the % character.

The character ? is accepted for the character %.

The following examples illustrate the use of wildcards:

Z%.BIN	All files beginning with Z and having extensions of BIN
Z%%%%%.BIN	All files beginning with Z and having extensions of BIN
.	All files regardless of filenames and extensions.

11.1 DIR Command

The directory command causes the directory of the specified device to be printed. Wildcards are allowed. The format of the command is: DIR DEV: (CR). If the DEV: is omitted, the directory will be obtained from the system device.

Command Examples:

DIR DEV:	Gives a directory of all files on DEV.
DIR *.BIN	Gives a directory of all files with a BIN extension. The system device is used.
DIR DEV:$.BI%	Gives a directory of all files with extensions beginning with BI.
DIR:ZYC##.BI%	Gives a directory of all files with the first three characters of the filename being "ZTC" and having extensions beginning with BI. The system device is used.

Example:

Entry#	FILNAM.EXT	DATE	LENGTH	START
000001	A .BIN	2-AUG-72	14	000105
000002	B .BIN	2-AUG-72	12C	000106

FILNAM.EXT is the name of the file.

Date is the file creation date.

Length is the number of blocks (in decimal) the file occupies. A "C" after the file length indicates the file is contiguous. Non-contiguous files are linked lists.

Start if the block number (in octal) of first block of the file.

The directory command accepts two switches. The "/F" switch gives a "FAST" directory containing only the entry and filename.

11.2 CLR Command

The CLR command sets to zero all locations in the buffer area used for loading/modifying/dumping files to/from a medium. The format of the command is: CLR(CR)

11.3 LOAD Command

The LOAD command loads files stored in ABS format into a buffer area within the utility program. Checksum tests are performed on the files. The format of the command is: LOAD DEV:FILNAM.EXT(CR)

If the DEV: is omitted, the file will be loaded from the system device.

If the device has no directory, then the file name and exension should be omitted. For example: LOAD PR:(CR) will load from the paper-tape reader.

Following the load, certain information is printed.

XFRADR: Indicates the starting address of the program loaded. If it is odd, the program is not self-starting.

CORE: Left number indicate the lowest location loaded into during the load. This value is known as the locore value. The right number indicates the highest location loaded into during the load. This value is known as the hicore value. The left and right numbers in effect indicate the core limits of the program. These core limits are relative to the buffer area within the utility.

Wildcards are allowed and will cause more than one file to be loaded. The files load into the same buffer in the utility program. The use of wildcards with this command is one way of performing checksum tests on a group of files.

11.4 DUMP Command

Contents of the buffer area within the utility can be written to a XXDP+ medium in ABS format by the dump command. The format of the command is: DUMP DEV:FILNAM.EXT.

If the DEV: is omitted, the file will be stored on the system device.

Processing starts from locore limit of the program and proceeds to but does not include the hicore limit of the program.

11.5 XFR Command

Once a program has been loaded into the buffer area within the utility via the load command, it can be made self-starting or not self-starting at the user's discretion. As described under load command, the load routine types: XFRADR: XXXXXX indicating whether a program is or is not self-starting. An example of XFR is:

XFR(CR)	Request current transfer Address
000001 000050	000001 is the current XFR address. 000050 is the new XFR address entered by the user.

11.6 SAVE Command

The save command writes the contents of the program buffer area of the utility program as a core image file. The transfer starts at the beginning of the buffer and ends at the word before the hicore value.

The only current use of the save command is to place a core image of the XXDP+ monitor on cassette and magtape. The format of the command is:

Save DEV:FILNAM.SAV(CR)

If the DEV: is omitted, the file will be stored on the system device.

11.7 MOD Command

Once a program is loaded with the load command it can be patched by the MOD command.

The format of this command is: MOD ADDRESS(CR)

The utility prints the address and its current contents. The user may type in an octal number and a terminator, or just a terminator.

If a number is typed, it is used as the new content of address.

The terminator can be either a carriage return or a line feed. A carriage return takes the program back to command mode, whereas a line feed causes the next word (ADR+2) to be opened for modification. For Example:

```
*MOD 50
000050  000005      3 (LF)
000052  012737      4 (LF)
000054  000300      5 (CR)
*MOD 50
000050  000003        (LF)
000052  000004        (CR)
```

NOTE: Line feed is ↓ as key pad of 2510 keyboard.

Any location within the locore/hicore limits can be modified with this command.

11.8 CORE Command

The core command causes the locore/hicore limits of the program buffer in the utility to be typed: The format of the command is: CORE(CR)

For Examples:

```
#CORE(CR)
00000,014776        Left number is the lower core limit,
                    Right number is the upper core limit
```

11.9 LOCORE Command

The locore command is used to change the lower limit of the program relative to the program buffer in the utility. The format of the command is: LOCORE ADR(CR). ADR is the new low core value.

11.10 HICORE Command

The Hicore command is used to change the upper limit of the program relative to the program buffer. The format of the command is: HICORE ADR(CR). ADR is the new high core value.

11.11 DELETE Command

The delete command causes a file to be deleted from the specified device. The format of the command is: DEL DEV:FILNAM.EXT(CR). The DEV: must be specified. Wildcards are allowed.

11.12 ZERO Command

The zero command causes all file information to be removed from the directory. The format of the command is: ZERO DEV:(CR). The DEV: must be specified.

If the specified device is the system device. A dialogue with the operator will occur. This discussion warns the operator than an additional read/write driver may be needed. A request to continue is given.

```
*ZERO DY0:

You may need additional driver
CONTINUE? (Y or N) Y
*
```

Note: Before entering Y to continue, the user places a scratch pack on the drive. Files from DY0 will be transferred to the scratch pack.

11.13 BOOT Command

The boot command causes block 0 of DEV to be loaded into memory. Starting at location 000000. Block 0 is assumed to have a boot loader. The program then jumps to location 000000 to start the boot loader. The format of the command is: BOOT DEV:(CR). If the DEV: is not specified, the system device will be booted.

11.14 SAVM Command

The SAVM command causes the first 4K of the buffer area in the utility program to be written as a core image. This command is used to write the XXDP+ monitor on the device as a core image that is bootable. The format of the command is: SAVM DEV:(CR)

This command is only valid for random access devices. The DEV: must be specified. For example:

```
#LOAD DY1:HMDY??.BIN   Load DYDP+ Monitor
#SAVM DY0:             Save it as core image on DY0:
```

Note: SAVM does not cause a directory entry.

The SAVM command also writes the first 256(10) words of the buffer into block 0 of the device. This is the boot loader.

11.15 REN Command

The REN command changes the name of a file on a medium. The format of the command is: REN DEV:NEWNAM.EXT=DEV:OLDNAM.EXT.

If the DEV: is not given, the system device will be used. Wildcards are allowed.

11.16 DRIVER Command

The driver command is used to load device drivers into memory. This command is not ordinarily required because drivers are loaded as part of the execution of the other update commands. For example, the command DIR DEV: causes the driver for DEV to be loaded and the directory of that device to be printed. However, the zero command creates a situation in which the user may need to have another driver loaded into memory.

The format of the driver command is: DRIVER DEV1: DEV2:.

11.17 PIP Command

The PIP command is used to transfer files from one medium to another medium. File data is not checked for format on checksums. The output file is given the date of the input file. The format of the command is: PIP DEV1:FILNAM.EXT=DEV2:FILNAM.EXT(CR)

The file names for the two devices can be different. If DEV is omitted, the system device will be used. Wildcards are allowed. Examples of this command follow:

PIP PP:=PR:	(Copies paper tape)
PIP DK0:ABC.BIN=PRI	Paper tape to disk
PIP PP:=ABC.BIN	Disk to paper tape punch
PIP DX0:*.*=*.BIN	Transfers all BIN files to DX0

The user should make sure that the output file name does not already exist in the output device directory.

Program files that have been PIPped to a XXDP+ device should be loaded immediately via the load command to ensure that no errors have occurred during the PIP command. ASCII files that have been PIPped can be printed via the type commands.

11.18 READ Command

The read command can be used to test for device errors by transferring files from a medium. The information obtained from the files is not retained. The format of the command is read DEV:FILNAM.EXT(CR). If DEV: is not given, the system device will be used. Wildcards may be used.

11.19 TYPE Command

The type command outputs files to the console terminal. It is used to print text files. The format of this command is: TYPE DEV:FILNAM.EXT.

If the DEV: is omitted, the system device will be used. Wildcards may be used.

11.20 DO Command

The DO command causes the execution of a command file. The file must be on the system device. The file can contain any legal UPD2 commands. The file is executed line by line and must not be terminated by an "Exit". Executable files are created via the XTECO text editor program. The format of the Do command is: DO DEV:FILNAM.EXT(CR).

11.20.1 Special Characters

Two characters, the semicolon (;) and the dollar sign ($), have special significance in the text file. The semicolon is used to start a comment which is to be printed during command file execution. The dollar sign is used to start a comment which is to be printed and followed by a halt during command file execution (such as $PRESS CONT WHEN READY).

11.21 ASG Command

The ASG (ASSIGN) command allows the use of logical device names in place of the actual or physical device name. Allowed logical device names are 0, 1, 2, 3, 4 and 5. The format of this command is:
ASG PHYSICALDEV:=LOGICALDEV:(CR).

11.22 PATCH Command

The patch command enables the user to patch a program on any block oriented (random access) XXDP+ supported device. No output DEV: file specification is required or permitted. The input device is assumed to be the desired output device.

The file to be patched must be in ABS format. The patch routine does not check in advance for correct file format.

The operator types "PATCH DEV:FILENAME", and UPD2 responds with the prompt "ADDR?" to which the operator should enter the address he wishes to patch. Then UPD2 will respond by typing the present contents of that location in the bin file and waiting for the operator to enter the desired new contents. When the operator does so, UPD2 responds by printing the "ADDR?" prompt again. The patches are all done in real time, rather than being accumulated in memory pending a verification response from the operator.

The operator may terminate the patch session by responding to the "ADDR?" prompt with a simple (CR).

11.23 EXIT Command

The exit command returns program control to the XXDP+ monitor. The format of this command is: EXIT (CR)

11.24 COPY Command

The copy command allows a file or an image copy to be performed between like devices. The operator enters the command "COPY NEWDEV:=OLDDEV:" and UPD2 responds by asking "FILE COPY OR IMAGE COPY?"

If the operator responds "I(MAGE)", then a check is made to ensure that this device is neither a bad block disk nor a tape. If this check is successful, then each physical block of the input device is written to the corresponding physical block of the output device. When the operation is completed, the output device is read. To perform the verification by checksum against the data that was read from the input device on the transfer pass.

If the operator responds "F(ILE)", then any device type is legal and the utility simply does the equivalent of a PIP out:*.*=IN:*.*.

11.25 UPD2 Errors

? INVALID DEVICE	Invalid device specified for command given.
? INVALID ADDRESS	Invalid address. Must be even and within existing locore and hicore limits.
? INVALID NAME	Invalid file name. No special characters allowed. A through Z, and 0 through 9 are only valid characters. Also occurs if * or wild character construction filenames are specified to a command that does not allow it.
? NON-EXISTENT FILE	Non-Existent file. File does not exist in device directory.
? DELETE OLD FILE	Delete old file before giving command that would create file with same name.
? RD/WT DEV ERR	Device Error on either input or output device. Check that output device is write-enabled.
? CHECKSUM ERROR	Checksum error during load command.
? END-OF-MEDIUM	End of medium. Occurs during input operations when the program attempts to input and the file is at an end. Serious problem. File in storage is probably wiped out.
? PROGRAM OVER-FLOW	Program too large to load within existing buffer space.
? INVALID COMMAND	The comand entered was not recognized by the utility programs.

11.25 UPD2 Errors (continued)

? "DO" FILE MUST BE ON SYSTEM DEVICE	The command file for the Do command must be on the system or load device.
? LOGICAL DEVICE NOT ASSIGNED	A command used a logical device mnemonic that had not been assigned to a physical device mnemonic via the ASG command.
? NO DEVICE DEFAULTS ALLOWED	The device mnemonic is missing for a command that requires it.

This page intentionally left blank.

Section 12

XTECO - TEXT EDITOR

12.0 Introduction

The XTECO - XXDP+ text editor program enables the user of XXDP+ to create and edit ASCII text files. All editing can be done by using a few simple commands.

XTECO is a character oriented editor. One or more characters in a line can be modified without retyping the rest of the line. XTECO does not require that line numbers be part of the text.

The input file for a given editing operation is the file to which the user wishes to make changes. If the user is using XTECO to create a new file, there is no input file. The output file is either the newly created file, or the edited version of the input file.

In general, the editing process proceeds as follows. The user specifies the file he wishes to edit, and then a block of text is read into core. The user modifies the text by using the various editing commands. He then appends additional blocks of text and edits them until the entire file has been edited. He next outputs the edited file and closes it.

The XETCO program shares memory with the XXDP+ monitor and uses certain functions made availble by the monitor. The exit command is used to return to the monitor.

XTECO does not contain read/write drivers for devices. These drivers are resident on the system device as SYS files. They are transferred to memory as they are needed. As a result, XTECO will run in 8K memory.

12.1 XTECO Command Description

The prompt for the XTECO command mode is the * character. The XTECO program provides several of the commands available under the update utility programs, in addition to those commands provided for editing purposes.

XTECO commands are of two types: Non-Edit and Edit.

The NON-EDIT commands are:

DIR	Update Equivalent
TYPE	Update Equialent
PRINT	Update Equivalent
(CR)	Carriage return is the non-edit type comand string terminator.
EXIT	Update Equivalent

The EDIT commands are:

TEXT	XTECO Unique
EDIT	XTECO Unique
TECO	XTECO Unique
(CTRL-Z)	Control-Z is the edit type command string terminator. Control-Z is not available on 2500 keyboards. Use EX.
I	Used to move pointer one or more lines
C	Used to move pointer one or more characters
J	Used to move pointer to beginning of text in core
ZJ	Used to move pointer to end of text in core
S	Used to search for a character sequence in text in core
N	Used to search core and remainder of input file for a specified character sequence
T	Used to type one or more text lines
D	Used to delete one or more characters
K	Used to delete (kill) one or more text lines
I	Used to insert ASCII text into the text buffer
A	Used to append one or more text blocks to text buffer
EX	Outputs edited file to output device and closes output
(ESC)	Echoes A $. Used to terminate an edit command
(ESC)(ESC)	Echoes 2 $. Used to terminate last edit command and to cause execution of entire command string.

NOTE

ESC may be ALT on some terminals.

The user should be aware of the use of the following special characters:

CTRL-C	Used to exit out of any command and return to command mode. Will cause an open output file to be closed. The user must be careful not to type CTRL-C, unless he wishes to abort his operation. This is specially true when editing a file, as all work will be wasted.
CTRL-O	Used to stop printing on the console terminal as when typing multiple lines of text when editing a file.
CTRL-U	Used to empty out contents of keyboard buffer, as when the user wishes to start typing his command sequence all over again.
RUBOUT or DELETE	Used to remove one or more characters typed from command or text string. One depression of the rubout key removes one character.

12.2 TEXT, EDIT, and TECO Commands

Text, edit and Teco commands are the basic commands provided to create or edit and ASCII text file.

The TEXT command is used when the user wishes to create a new text file. The TEXT command does not require an input file, only an output file. All editing commands are available with the exception of the (append) command which becomes a no-op command when no input file exists.

The EDIT command is the general purpose command for editing an existing text file. It permits the user to edit and input file on one type of device and to output the edited file to a different type device. All editing commands are available when under the edit command.

The TECO command is a specialized version of the edit command. Under the TECO command the input and output device/drive must be the same (it assumes they are the same), and must be random access type devices (disk, floppy). The edited output file takes its name from the name and extension of the input file, and the input file is renamed to a BAK extension (for backup). All editing commands are availble.

Individual command descriptions follow.

12.2.1 TEXT Command

The text command is used to create a new text file. The format is:

TEXT OUTDEV:FILNAM.EXT(CR)

Where OUTDEV: is any directory device. If it is omitted, the system device is used.

The program will type:

Make output ready. Type (CR) when ready.

Ensure that the output device is ready and write-enabled. Press the (CR) key on the console terminal when ready to proceed. The program is now in edit mode, and only editing type commands are valid. The program prompts the user by typing quote (").

The user can at this point type and edit his text. (ESC)(ESC) EX (ESC) is used to return to XTECO command mode and write the new text as a file on the medium.

12.2.2 EDIT Command

The edit command permits the user to edit a text file from a specified input device, and to output the edited text file to a specified output device. The command format is:

EDIT OUTDEV:FILNAM.EXT=INDEV:FILNAM.EXT(CR)(CR)

Both OUTDEV: and INDEV: must be directory devices. If they are omitted, the system device is used.

The program will type:

Make output ready. Type (CR) when ready.

Ensure that the output device is ready and write enabled. Press the (CR) key on the console terminal when ready to proceed. The program is now in edit mode, and only editing type commands are valid. The program prompts the user by typing quote (").

The user can at this point type and edit his text. (ESC)(ESC) EX (ESC) is used to return to XTECO command mode and write the new text as a file on the medium.

12.2.3 TECO Command

The TECO command is a specialized, short hand version of the edit command. When using the TECO command the input device/drive must be the same as for output. In addition, the command is reserved for use only with random access devices. The command format is:

TECO DEV:FILNAM.EXT(CR)

Where DEV: is any random access device. If DEV: is omitted, the system device is used.

It is important that the user be aware of the mechanics involved in the operation of the TECO command. The sequence is as follows:

1. Open input file.
2. Open output file, add assign it a .TMP extension.
3. Edit operations are performed by operator.
4. Editing done. Output edited file to .TMP file.
5. Close the .TMP file.
6. Rename the .TMP file to same name and extension as the input file.
7. Rename the input file to a BAK extension.

The input device must not contain a file with the same name as the input file and BAK extension. It will be deleted in the process of renaming the input file to a BAK extension.

Note that the TECO command must not be used to edit a file which has the extension BAK. The file must first be renamed to another extension.

The program will type:

Make output ready. Type (CR) when ready.

As shown in the above example, the escape used to terminate an alphanumeric argument may also serve as one of the two altmodes necessary to terminate a command string.

12.3 XTECO Edit Commands

12.3.1 A (APPEND) Command

The A command reads in the next block of text from the input device and adds it to the contents of the text buffer in core.

12.4 Buffer Pointer Positioning Commands

Since XTECO is a character-oriented editor, it is very important that the user understand the concept of the buffer pointer. The position of the buffer pointer determines the effect of many of the editing commands. For example, insertion and deletion always takes place at the current position of the buffer pointer.

The buffer is the current text contents in core, from the first character, up to and including the last character.

The buffer pointer is simply a movable position indicator. It is always positioned between two characters in the buffer, or before the first character in the buffer, or after the last character in the buffer. The pointer may be moved forward or backward over any number of characters.

12.4.1 J Command

The J command moves the pointer to the beginning of the buffer, i.e., immediately before the first character in the buffer.

12.4.2 ZJ Command

The ZJ command moves the pointer to the end of the buffer, i.e., to position following last characters in the buffer.

12.4.3 C Command

The C command moves the pointer one character in the buffer. The C Command may be preceded by a (decimal) numeric argument. The command NC moves the pointer forward over N characters. The command -NC moves the pointer backward over N characters. (The pointer cannot be advanced beyond the ends of the buffer).

12.4.4 L Command

The L command is used to advance the buffer pointer on a line-by-line basis. The L command takes a numeric argument, which may be positive, negative, or zero, and is understood to be 1 if omitted.

Ensure that the output device is ready and write-enabled. Press the (CR) key on the console terminal when ready to proceed. The program is now in edit mode, and only editing type commands are valid. The program prompts the user by typing quote (").

The user can at this point type and edit his text. (ESC)(ESC) EX (ESC) is used to return to XTECO command mode and write the new text as a file on the medium.

12.5 Editing Command String Syntax

XTECO commands may be given one at a time. However, it is usually more convenient to type in a command string containing several commands that form a logical group. An example of a command string is shown below.

"IHEADING$NTAG:$2LT$$	Inserts word heading, searches for string tag:, moves pointer forward 2 lines and types line pointed to.

Command strings are formed by typing one command after another, separated by one escape. Command strings are terminated by typing two consecutive escapes.

Execution of the command string begins only after the double escape has been typed. At that point, each command in the string is executed in turn, starting at the left. When all commands have been executed, XTECO prints another quote, indicating readiness to accept another command.

If some commands in the string cannot be executed because of a command error, execution of the command string stops at that point, and an error message is printed. Commands preceding the bad command are executed. The bad command and those following it are not executed.

12.6 Command Arguments

There are two types of arguments for XTECO editing commands. Some commands require numeric arguments and other commands require alpha-numeric (text) arguments. Numeric arguments are decimal integers. Numeric arguments always precede the command to which they apply. A typical example of a command taking a numeric argument is the command to delete three characters: 3D.

Alpha-numeric arguments are textual arguments meant to be interpreted as ASCII code by XTECO. Alpha-numeric arguments always follow the command to which they apply, and they must always be terminated by an altmode. Examples of alpha-numeric arguments are (1) text to be inserted, and (2) character strings to be searched for.

Example:

*ISOMETHING$$	:The argument is something

Suppose the buffer pointer is positioned at the beginning of line 8 or at some position within line 8.

The command NL, where N >0, advances the pointer to the beginning of line 8+N.

The command -0L moves the pointer to the beginning of Line 8. If the pointer is already at the beginning, nothing happens.

The command -NL moves the pointer back to the beginning on line 8-N.

NOTE: Execution of the A (APPEND) command does not change the position of the buffer pointer.

12.7 Text Type-Out Commands

12.7.1 T Command

Various parts of the text in the buffer can be typed out for examination by use of the T command. Just what is typed out depends on the position of the buffer pointer and the argument given. The T command never moves the buffer pointer.

The T command types out everything from the buffer pointer through the next line feed. Thus, if the pointer is at the beginning of a line, the T command causes that line to be typed out. If the pointer is in the middle of a line, T causes the portion of the line following the pointer to be typed.

The Command NT (N >0) is used to type out N lines, i.e., everything from the buffer pointer through the nth line feed following it.

The user, especially one new to XTECO, should use the T command often to make sure the buffer pointer is where he thinks it is.

During execution of any T command, the user may stop the terminal output by typing the CTRL-O character. The printing stops and execution of the remainder of the command string is aborted. Therefore, long typeouts should be restricted to single command, command strings.

12.8 Deletion Commands

12.8.1 D Command

Individual characters are deleted by using the D command. The command D deletes the character immediately following the buffer pointer. The command ND, where N >0 deletes the 0 characters immediately following the pointer.

12.8.2 K Command

Lines are deleted by using the K command. The K command may be preceed by a numeric argument, which is understood to be a 1 if omitted. The command NK (N >0) deletes everything from the current position of the buffer pointer through the Nth line feed character following the pointer.

At the conclusion of a D or K command the buffer pointer is positioned between the characters which precede and follow the deletion.

12.9 Insertion Command

The only insertion command is the I command. The ASCII text that is to be inserted into the buffer is typed immediately after the letter I. The text to be inserted is terminated by an altmode.

Any ASCII character except null, altmode, rubout, CTRL-C, CTRL-O, and CTRL-U may be included in the text to be inserted.

If a carriage return is typed in an insertion, it is automatically followed by a line feed. The text to be inserted is placed in the buffer at the position of the buffer pointer, i.e., between the characters. At the conclusion of the insertion command the buffer pointer is positioned at the end of the insertion.

Any number of lines may be inserted with a single I command. However, it is recommended that no more than 10 to 20 lines should be inserted with each I command.

12.10 Output Command

The only output command available with XTECO is the EX (EXIT) command. The EX command is used to conclude an editing job with a minimum of effort. Its use is best shown by an example:

Suppose the user is editing a 30 page file and suppose the last actual change to the file is made on page 10. At this point the user give the command:

```
EX$$
*
```

The action of XTECO is (1) to rapidly move all of the rest of the input file to the output file, (2) close the file, and (3) to return to command mode so that the user may give other NON-EDIT mode commands.

12.11 Search Commands

In many cases the simplest way to position the buffer pointer is by using a character string search. A serach command causes XTECO to search through the test until a specified string of characters is found, and then to position the pointer at the end of this string. There are two search commands.

12.11.1 S Command

The S command is used to search for a character string within the buffer. The string to be searched for is specified as an alpha-numeric argument following the S command. This argument must be terminated by an escape.

Execution of the S command begins at the position of the buffer pointer and continues to the end of the buffer. If the specified string is not found an error message is printed and the buffer pointer is set to the point where the search began.

12.11.2 N Command

The N command works just like the S command. The difference is that an S command ends at the end of the buffer, whereas the N command does not. An N search begins like an S Search but if the character string is not found in the current buffer an automatic A (Append) command is executed and the search continued until the search is successful or the input file exhausted.

If the N command finds the specified string the pointer is positioned at the end of the string found. If the string is not found, an error message is printed and the pointer is set at the beginning of the buffer. Since a good part of the file may already have been output to the output device, the user may have no other choice than to exit via the EX command, and to reopen the file and try the N search again with a character string that can be found.

CAUTION

When attempting to search it is very easy to overlook an occurrence of the search string preceding the one the user desires. For example, he may want to move the pointer after the word "and" but erroneously position the pointer after a preceding occurrence of a word like "thousand".

For this reason, the user is strongly urged to execute a T command to ascertain the position of the pointer after each search command.

12.12 XTECO Errors

Error messages generated by XTECO are included in those generated by the UPDATE program. Refer to the descriptions given for the UPDATE programs.

In addition, one error message is generated by XTECO when a search for a character string by either the S or N commands fails. In that case XTECO types:

Not Found: (ASCII String)

This page intentionally left blank.

Section 13

HELP TEXT

13.0 XXDP+ Help Text File (HELP.TXT)

13.1 XXDP+ Monitor

13.1.1 XXDP+ Monitor Commands:

R FILENAME ADDR	Loads and starts program at specified address (address defaults to XFR address of program)
L FILENAME	Load Program
S ADDR	Start Program at specified address (address defaults to XFR address of program)
C FILENAME?QV	Executes chain file (/QV gives optional quick verify mode)
D/L/F	Directory (optionally on printer (/L) or in short form (/F)
F	Set console fill count
E UNIT #	Enable alternate system device
H/L	Help file (optionally on printer (/L))

NOTE

The XXDP+ monitor can run either the old or the new XXDP utilities, but the old XXDP monitor cannot run the new utilities.

The new monitor is a 2K program (unlike the old XXDP monitor, which as a 1.5K program). However, all old diagnostics are still supported, regardless of size.

13.1.2 Boot Time Questions

The monitor will ask you at boot time whether you have 50 Hz and whether you have an LSI. This is information that it cannot autosize for and which it must pass on to certain new types of diagnostics.

Both questions default to "NO" if you type a (CR).

The monitor also asks the date at boot time. You can take the default of 1-Jan-70 by typing (CR).

NOTE

XXDP+ utilities use the date supplied to the monitor at boot time.

13.1.3 Enable Command

The enable command has been upgraded to handle any device (formerly handled just cassettes). If you are booted from one drive and want the monitor to behave as if it had been booted from another, just type "E UNIT #" (e.g., "E 2" for Drive #2).

13.1.4 Directory Utility

The special utility HUDI??.SYS is autoloaded by the monitor when the "D" command is issued by the operator. It performs the directory function and overplays the contents of memory (except the monitor).

13.2 XXDP+ Utilities

13.2.1 XXDP+ UPD1: Utility Commands

CLR	Clear Memory
XFR	Set Transfer Address
MOD ADDR	Modify Core
LOAD DEV:FILENAME	Load Program
DUMP DEV:FILENAME	Dump Program
CORE	Display Core Limits for Dump
HICORE ADDR	Set hicore limit for dump
LOCORE ADDR	Set locore limit for dump
DEL DEV:FILENAME	Delete Program
BOOT DEV:	Boot Device

NOTE

The UPD1 utility supports all standard XXDP devices, but only the above command set (for example, note that the "pip" and "start" commands are no longer available.

UPD1 needs to be used only in cases where there is not sufficient memory to allow the LOAD/MOD/DUMP operation on a binary file. Otherwise, UPD2 may be used.

It is a 4K program and it executes in low core (where it loads). It destroys the monitor image in high core (it uses that area as a buffer). Therefore, you must reboot the monitor when finished.

The load device must remain online throughout execution, since UPD1 uses retrievable device drivers from that medium as it executes.

13.2.2 XXDP+ UPD2 Utility Commands

CLR	Clear Memory
ZERO DEV:	Zero Device
SAVM DEV:	Save Monitor to Disk
SAVE DEV:FILENAME	Save Monitor to Tape
XFT	Set Transfer Address
LOAD DEV:FILENAME	Load Program
DUMP DEV:FILENAME	Dump Program
CORE	Display Core Limits for Dump

13.2.2 XXDP+ UPD2 Utility Commands (continued)

HICORE ADDR	Set hicore limit for dump
LOCORE ADDR	Set locore limit for dump
DIR DEV:FILENAME	Directory (optionally for a particular file)
PATCH DEV:FILENAME	Patch Disk File
ASG PHDEV: LOGDEV	Assign Logical Name
READ DEV:FILENAME	Read File for Validity
PIP DEV:FILENAME=DEV:FILENAME/F	Transfer file. The /F switch (for "fast") causes UPD2 to write files to tape without checking to see whether they are already present, thus potentially duplicating the name
EOT DEV:	Write end of tape mark
DEL DEV:FILENAME	Delete File
REN DEV:FILENAME=DEV:FILENAME	Rename file
DO FILENAME	Execute command file. It must be on system device and it must not contain "exit" CMD
DRIVER DEV:	Load R/W Driver into Core
BOOT DEV:	Boot System
PRINT DEV:FILENAME	Print File (On LPT)
TYPE DEV:FILENAME	Type File (on TTY)
MOD ADDR	Modify Core
Copy DEV:=DEV:	Copy volume (need like devices)
EXIT	Return control to monitor

NOTE

The UPD2 utility supports all standard XXDP devices, but the "start" command is no longer supported (necessary to use monitor for that).

It is a 6K program that executes in low core (where it loads) and makes calls to the monitor as it executes. The exit command returns control to the monitor.

UPD2 uses retrievable I/O drivers from the system device as it executes, therefore that device must remain online during its execution.

13.2.3 PIP/FILE Rules

Here's what happens when you do a "PIP" or a "FILE" between disks or tapes, and the file being transferred is already present on the output device ("D" and "T" stand for disk and tape):

	Single File Mode	Wild Card Mode
FILE	D: Says "Delete Old" T: Says "Delete Old:	D: Autodeletes T: Autodeletes
PIP	D: Says "Delete Old" T: Says "Delete Old:	D: Autodeletes T: Autodeletes
FILEF	D: Says "Delete Old" T: Duplicates the name	D: Autodeletes T: Duplicates the name

PIP/F	D: Says "Delete Old" T: Duplicates the Name	D: Autodeletes T: Duplicates the Name

NOTE

The reason that the "fast mode commands" (the "FILEF" command and the "PIP" command with the "/F" switch) create duplicate name on tape is that the output tape is not rewound to search for the presence of a file with the same name as that of the specified output file.

The "FILE: and "FILEF" commands, although still supported, have been officially replaced by the "PIP" and "PIP/F", which are functionally identical (although not syntactically identical: the "PIP" needs output filename, which the "FILE" doesn't take).

13.2.4 XXDP+ XTECO Utility Commands

DIR DEV:FILENAME	Directory (optionally of specific file)
PRINT DEV:FILENAME	Print file (on LPT)
TYPE DEV:FILENAME	Type File (on TTY)
EDIT DEV:FILENAME=DEV:FILENAME	Edit ASCII File
TECO DEV:FILENAME	Edit single file in place
TEXT DEV:FILENAME	Create new text file
C	Move pointer by characters
L	Move pointer by lines
J	Move pointer to start of text in memory
ZJ	Move pointer to end of text in memory
S	Search for character sequence
N	Search to end of file for sequence
T	Type lines of text
D	Delete characters
K	Delete lines of text
I	Insert text
A	Append more text from file into memory
(ESC)	Terminate a command
(ESC)(ESC)	Terminate last command and execute any command
-C	Return to command mode
-U	Restart command sequence
RUBOUT	Ignore last character
-Z	Terminate input mode
EXIT	Return control to monitor

13.2.5 Zeroing a System Device

If you attempt to zero the system load device, the utility program will warn you to load another device driver. This warning is given because device drivers reside as SYS files on the system device. Before you zero the system device, you must have two device drivers in memory: one for the system device and one for the device you will use to rebuild the system device. The zero command provides the first driver. The "driver" command can be used to obtain the second driver. If both devices are of the same type, then only one driver is needed.

13.3 Miscellaneous Operations

13.3.1 Non-Standard CSR

If your system device has a non-standard CSR, patch location 20 of the monitor and location 24 of the read/write driver to the correct value. In patching the monitor, it is necessary to patch the ".SYS" file and then to do a "SAVM" to the disk.

13.3.2 Chaining Utilities

The monitor chain command can be used to execute the UPD2 utility as follows. Let the chain file contain the command "R UPD2". After this, place whatever UPD2 commands you wish to have executed. At the end of these, put the UPD2 command "Exit". You may then follow with any other legal monitor commands.

The monitor will only chain files which end with the extension "BIC", so it is necessary to copy or rename your "UPD2.BIN" to UPD2.BIC before chaining.

13.3.3 How to Merge XXDP+ Software onto Your Disk

a) Boot from old XXDP pack
b) Run old UPD2
c) File OLDEV:=NBEWDEV:*.SYS (this brings across the new monitors and the new drivers and the directory utility HUDI??.SYS which is automatically invoked by the monitor on a "D" command)
d) File OLDEV:=NEWDEV:*.BIN (This brings across the new UPD2, UPD1, XTECO, and DXCL utilities)
e) File OLDEV:=NEWDEV:HELP.TXT (Brings across this file)
f) LOAD NEWDEV:HMXX??.SYS (This is the new monitor for your system device: the "XX" is the device mnemonic, as explained below)
g) SAVM OLDEV: (Puts new monitor image in boot area of old pack)
h) You are still running the old UPD2: Now boot your device and you'll be running the new monitor.

13.3.4 How to Merge Your XXDP Tape onto an XXDP+ Tape

a) Boot from new tape
b) Run new UPD2
c) Delete OLDEV:THDP.SAV
d) Delete OLDEV:TMDP.SAV
e) Delete OLDEV:UPD1.BIN
f) Delete OLDEV:UPD2.BIN
g) Delete OLDEV:XTECO.BIN
h) FILEF NEWDEV:=OLDEV:*.*

13.3.5 How to Build XXDP+ Disk From Scratch

a) Zero the disk
b) Load monitor for that disk type
c) Do a "SAVM" to that disk
d) Transfer all desired files to that disk

13.3.6 How to Build XXDP+ Tape From Scratch

a) Zero the tape
b) Load MM monitor
c) Do a "SAVE" of the MM monitor to the tape (EXT = "SAV")
d) Load the MT monitor
e) Do a "SAVE" of the MT monitor to the tape (EXT = "SAV")
f) Transfer desired files to the tape

13.3.7 Loading an "Oversize" Diagnostic

The XXDP+ monitor is a 2K program (XXDP was 1.5K). The lower .5K is "expendable" and is restored by the rest of the monitor when it is found to be missing (by checksum).

It is possible to run a 14.5K diagnostic on a 16K machine. Such a diagnostic, however, should not be loaded and started by two separate commands load/start. This is because the TTY I/O routines are in the lower .5K of the monitor and if the diagnostic overlays that area you won't be able to talk to the keyboard. Therefore if the monitor detects that you have issued the "load" command for such a file, it will type the message "Continue Load Thereby Disabling Keyboard?" If you answer "yes", it will complete the load, but you will have to start the diagnostic via console switches. If you answer "no" the monitor returns to command mode, and you may issue a "run" command for that diagnostic (with an optional starting address) and you won't be affected by the lower .5K being temporarily overlaid.

13.4 XXDP+ Naming Conventions

13.4.1 XXDP+ Monitor and Device Driver Naming Conventions

Device Mnemonic	Monitor File	Driver File	Devices Supported
CT	HMCT??.SYS	HDCT??.SYS	TA11/TO60
DB	HMDG??.SYS	HDDB??.SYS	RP01/5/6
DD	HMDD??.SYS	HDDD??.SYS	DL11/TO58
DK	HMDK??.SYS	HDDK??.SYS	RK11/RK05
DL	HMDL??.SYS	HDDL??.SYS	RL11/RL01
DM	HMDM??.SYS	HDOM??.SYS	RK611/RK06/7
DP	HMDP??.SYS	HDDP??.SYS	RP11/RP02/3
DR	HMDR??.SYS	HDDR??.SYS	RM02/3
DS	HMDS??.SYS	HDDS??.SYS	RS03/4
DT	HMDT??.SYS	HDDT??.SYS	TC11 DECTAPE
DX	HMDX??.SYS	HDDX??.SYS	RX11/RX01

13.4.1 XXDP+ Monitor and Device Driver Naming Conventions (continued)

Device Mnemonic	Monitor File	Driver File	Devices Supported
DY	HMDY??.SYS	HDDY??.SYS	RX211/RX02
MM	HMMM??.SYS	HDMM??.SYS	TM02/TM03 Tapes
MT	HMMT??.SYS	HDMT??.SYS	TH11 Tapes
KB		HDKB??.SYS	PPT RDR (LOW SP)
PT		HDPT??.SYS	PPT PCH (LOW SP)
PR		HDPR??.SYS	PPT RDR (HIGH SP)
PP		HDPR??.SYS	PPT PCH (HIGH SP)

NOTE

The device mnemonics are unchanged from previous XXDP, but all other names are now derived from these mnemonics. All monitors start with "HM" and all drivers start with "HD". The ?? in each file is the Rev/Patch level.

Files with the extension ".SYS" are binary files that cannot be directly executed by the operator.

Tape monitors must be given the extension ".SAV" when placed at the start of a magtape for booting, and ".SYS" when residing in the normal file area on either tape or disk.

13.4.2 XXDP+ Utility Naming Conventions

The file names on the XXDP+ utilities are as follows:

```
UPD1.BIN     UPDATE UTILITY #1
UPD2.BIN     UPDATE UTILITY #2
XTECO.BIN    TEXT EDITOR
DXCL.BIN     DEC/X11 CONFIGURATOR LINKER

END OF HELP.TXT
```

This page intentionally left blank.

Section 14

SUPERVISORY DIAGNOSTICS

14.0 Operating Instructions

14.1 How to Run

14.1.1 The Five Steps of Execution

A supervisor diagnostic should be started using normal XXDP procedures.

When the diagnostic is started, the following steps will occur:

Step 1

When the diagnostic issues the prompt "DR ", you will enter a "START" command. This is not the same as the XXDP "START" command, which you already issued in response to the XXDP dot prompt. This "START" command can take a number of switches and flags (all optional) and the details of these are set forth in 5.2 Details of Commands and Syntax. However, in order to use the program, all you need to say is something like this:

```
STA/PASS:1/FLAGS:HOE
```

Things to note here:

1. Only the first three characters of this or any command at the "DR " level need to be typed.

2. The "PASS" switch specifies how many passes you desire. A pass consists of running the full diagnostic against all units being tested (this will be explained shortly). One pass is specified in the above example.

3. The "FLAGS" switch may specify any of a number of flags, but the main useful ones are:

LOE	Loop on error
HOE	Halt on error
IER	Inhibit error printout

The HOE flag is specified in the above example.

Step 2

When you have typed in a "START" command, the diagnostic will come back with the question "$ UNITS?" to which you should respond by typing in the number of devices you wish to test.

Note that the number of units depends on the target device of the diagnostic. For example, if the diagnostic is directed at a disk drive, the number of

units would be the number of drives to be tested. Whereas if the diagnostic was directed at the disk controller, then the number of units would be the number of controllers. The target device of a diagnostic can always be determined by inspecting the "HEADER" statement near the beginning of the source code. One of the operands of this "HEADER" statement should be the device type of the diagnostic.

Step 3

When you have typed in the number of units to be tested, the diagnostic will ask you the "HARDWARE QUESTIONS". The answers to these questions are used to build tables in core, called hardware P-tables. One hardware P-table will be built for each unit to be tested.

There are several hardware questions and the entire series will be posed N times, where N is the number of units.

This represents a new philosophy in diagnostic engineering. Diagnostic will in the future not be written to autosize or assume standard addresses. Instead, they will ask the operator for all the information they need to test the device.

Step 4

After you have answered all the hardware questions for all the units, you will be asked "CHANGE SW?". If you want to be asked the software questions that determine the behavior of this program, type "Y". If you want to take all the defaults to these questions, type "N". If you type "Y" you will be asked the software questions, and the answers will be put into the software P-table in the program. The series of questions will be asked just once, regardless of the number of units to be tested.

Step 5

After you have answered the software questions, the diagnostic will begin to execute the hardware test code. There are several things that can happen next, depending on whether a hardware error is encountered and also on what switch values you selected on the START command. Consider the possibilities:

1. If no error is encountered, then the diagnostic will simply execute the desired number of passes and return to command mode (prompt DR).

2. If an error is encountered, then one of three things happens, depending on the settings of the HOE and LOE flags.

   HOE set: The error will be reported on the console and the diagnostic will return to command mode.

   LOE set: The diagnostic will loop endlessly on the block of code that detected the error.

   Neither HOE nor LOE set: The error will be reported on the console and normal execution will resume as if no error had occured.

14.2 Details of Commands and Syntax

14.2.1 Table of Command Validity

There are four ways of entering diagnostic command mode, and different subsets of the diagnostic command set are available with each:

	How Entered	Legal Commands
1.	Operator entered "RUN DIAG"	START PRINT DISPLAY FLAGS ZFLAGS EXIT
2.	Diagnostic has finished all requested passes	START RESTART PRINT DISPLAY FLAGS ZFLAGS EXIT
3.	Operator interrupted the diagnostic with CTRL/C	START RESTART CONTINUE PRINT DISPLAY FLAGS ZFLAGS EXIT
4.	An error was encountered with the HOE flag set set	START RESTART CONTINUE PROCEED PRINT DISPLAY FLAGS ZFLAGS EXIT

This page intentionally left blank.